Chapter5:Proof systems for ASL

1 Introduction

In this chapter, we present different consequence relations which are sound and in some cases complete with respect to the consequence relation between sentences and specifications of different $ASLker$ and $ASLnf$ languages and a proof system for the refinement relation between specifications presented in the chapter of the semantics of ASL. The consequence relations will be defined in terms of proof systems in order to give adequate representations of the consequence relations in the type theory UTT following the techniques presented in chapter 3. Most of the consequence relations will be presented for a given $ASLker$ specification language in some cases with an arbitrary but fixed algebraic institution and in others with the concrete algebraic institutions $FOLEQ$ and HOL presented in the previous chapter. For the latter, we will also present consequence relations for the institutions of $FOLEQ$ and HOL by means of proof systems.

The judgements which we will use for the definition of the proof systems for the algebraic institutions $FOLEQ$ and HOL are:

- ϕ. This judgement means that *the formula ϕ holds.*
- $\phi : \tau$. This judgement will be used for the definition of Π_{HOL} and it means that *the higher-order formula ϕ has type τ.*
- $\phi =_{\beta, X_{FOLEQ}} \phi$. This judgement will be used for the definition of Π_{HOL} and it means that *the higher-order formulas ϕ and ϕ' are equivalent by β-equality.*

The proof system for $FOLEQ$ ($\Pi_{FOLEQ}(\Gamma, \Sigma)$) for any set of sentences Γ with signature Σ will be formulated as a natural deduction system with the only sequent $\Gamma \Rightarrow_{X_{FOLEQ}} \phi$ where in this case X_{FOLEQ} is a finite set of pairs of variable and its associated sort and the proof system for HOL ($\Pi_{HOL}(\Gamma, \Sigma)$) for any set of assumptions Γ with signature Σ will be formulated with the following sequents:

- $\Gamma \Rightarrow_{X_{HOL}} \phi$ where in this case X_{HOL} is is a finite set of pairs of variable and its associated higher-order type
- $X_{HOL} \blacktriangleright \phi : \tau$ where ϕ is a higher-order formula and τ its associated type in $Types_{HOL}(\Sigma)$.
- $\phi =_{\beta, X_{HOL}} \phi'$ where ϕ and ϕ' are higher-order formulas.

In the following definitions, $INSFH$ will range over $FOLEQ$ and HOL:

Definition 1.1 *The consequence relation* $\vdash_{\Sigma, \Pi_{INSFH}(\Gamma,\Sigma)}$: $\mathcal{P}(|Sen_{INSFH}(\Sigma)|) \times |Sen_{INSFH}(\Sigma)|$ *for a given* $\Sigma \in |AlgSig|$ *is defined as follows:*

$$\Gamma \vdash_\Sigma \phi \Leftrightarrow \textit{ Any closed sequent of the form } \Gamma \Rightarrow_X \phi$$

$$\textit{has a derivation in } \Pi_{INSFH}(\Gamma, \Sigma).$$

Definition 1.2 *A consequence relation* $\vdash_{\Sigma,\Pi_{INSFH}(\Gamma,\Sigma)}$: $\mathcal{P}(|Sen_{INSFH}(\Sigma)|) \times |Sen_{INSFH}(\Sigma)|$ *is sound if it satisfies the following condition:*

$$\forall \phi \in Sen_{INSFH}(\Sigma).\Gamma \vdash_{\Sigma,\Pi_{INSFH}(\Gamma,\Sigma)} \phi \Rightarrow$$

$$(\bigwedge_{\psi \in \Gamma} A \models_{\Sigma,INSFH} \psi) \Rightarrow (A \models_{\Sigma,INSFH} \phi)$$

Definition 1.3 *A consequence relation* $\vdash_{\Sigma,\Pi_{INSFH}(\Gamma,\Sigma)}$: $\mathcal{P}(|Sen_{INSFH}(\Sigma)|) \times |Sen_{INSFH}(\Sigma)|$ *is sound and complete if it satisfies the following condition:*

$$\forall \phi \in Sen_{INSFH}(\Sigma).\Gamma \vdash_{\Sigma,\Pi_{INSFH}(\Gamma,\Sigma)} \phi \Leftrightarrow$$

$$(\bigwedge_{\psi \in \Gamma} A \models_{\Sigma,INSFH} \psi) \Rightarrow (A \models_{\Sigma,INSFH} \phi)$$

The proof systems which are used in the definition of consequence relations can be divided in non-compositional and compositional proof systems. The non-compositional proof systems are inductively defined by specification expressions for $ASLker$ specifications languages assigning for every case specific rules and axioms, and for $ASLnf$ specification languages, these proof systems are defined for the normal forms of the specifications. The compositional proof systems are defined by a single set of rules in such a way that the derivations of the proof systems preserve the structure of the specification.

Our usual denotation of these proof systems will be Π^{ASL}_{AINS} where $AINS$ is a concrete or an arbitrary institution and ASL is an $ASLker$ or an $ASLnf$ specification language. If the definition of the proof system is not the same for all the specification expressions of the language, we will denote by $\Pi^{ASL}_{AINS}(SP)$ the proof system associated to the specification expression SP of ASL.

Finally, we present a proof system for the refinement of specifications of a concrete $BASLker$ specification language which we will refer to as $RBASL$ with a fixed but arbitrary algebraic institution.

2 Non-compositional proof systems

2.1 Inductively defined by specification expressions

In this subsection, we present this kind of proof systems for a $BASLker$ specification language which we will denote as $RBASL$ with the algebraic institutions $FOLEQ$ and HOL. $RBASL$ includes a renaming and a reachability operator appart from the common operators of $BASLker$ languages. We present first the language $RBASL$ and then the proof systems and consequence relations associated to these institutions.

Definition 2.1 *A reachability constraint for a given signature* $\Sigma = (S, Op)$ *is a pair of a set of sorts and a set of functions* $(\mathcal{S}_{\mathcal{R}}, \mathcal{F}_{\mathcal{R}})$ *such that* $\mathcal{S}_{\mathcal{R}} \subseteq S$ *and for any* $f : s_1 \times \ldots \times s_n \to s \in \mathcal{F}_{\mathcal{R}}$, $s \in \mathcal{S}_{\mathcal{R}}$.

Definition 2.2 *For any signature $\Sigma = (S, Op) \in AlgSig$, an algebra $A \in Alg(\Sigma)$ satisfies a reachability constraint $(\mathcal{S}_\mathcal{R}, \mathcal{F}_\mathcal{R})$ of Σ ($A \models (\mathcal{S}_\mathcal{R}, \mathcal{F}_\mathcal{R})$) if the following condition holds:*

$$A \models (\mathcal{S}_\mathcal{R}, \mathcal{F}_\mathcal{R}) \Leftrightarrow \forall s \in \mathcal{S}_\mathcal{R}.\forall v \in A_s.$$

$$\exists t \in T_{(S,\mathcal{F}_\mathcal{R})}(X_{S-\mathcal{S}_\mathcal{R}}).\exists \alpha : X_{S-\mathcal{S}_\mathcal{R}} \to A.I_\alpha(t) = v$$

Definition 2.3 *RBASL is a BASLker language with additionally the following operators:*

$$Signature(rename \quad SP \quad \textbf{by} \quad \sigma) = \Sigma$$

$$Symbols(rename \quad SP \quad \textbf{by} \quad \sigma) = \sigma'(Symbols(SP))$$

$$Models(rename \quad SP \quad \textbf{by} \quad \sigma) = \{ A \in Alg(\Sigma) | A|_\sigma \in Models(SP) \}$$

$$Signature(\textbf{reach} \quad SP \quad \textbf{with} \quad (\mathcal{S}_\mathcal{R}, \mathcal{F}_\mathcal{R})) = Signature(SP)$$

$$Symbols(\textbf{reach} \quad SP \quad \textbf{with} \quad (\mathcal{S}_\mathcal{R}, \mathcal{F}_\mathcal{R})) = Symbols(SP)$$

$$Models(\textbf{reach} \quad SP \quad \textbf{with} \quad (\mathcal{S}_\mathcal{R}, \mathcal{F}_\mathcal{R})) = \{A \in Models(SP) \mid A \models (\mathcal{S}_\mathcal{R}, \mathcal{F}_\mathcal{R})\}$$

where $\sigma : Signature(SP) \to \Sigma$ is a bijective signature morphism and σ' is the pushout morphism of the following diagram:

$$\begin{array}{ccc} Sym(SP) & \xrightarrow{\sigma'} & PO(i,\sigma) \\ {\scriptstyle i}\uparrow & & \uparrow \\ Sign(SP) & \xrightarrow{\sigma} & \Sigma \end{array}$$

where i is the inclusion $i : Signature(SP) \hookrightarrow Symbols(SP)$.

Similarly to the consequence relations associated to the proof systems for $FOLEQ$ and HOL, we can define consequence relations of sentences of $FOLEQ$ and HOL from specification expressions using non-compositional proof systems. The definition of these consequence relation will use a function $Symbols$, which given a proof system Π^{ASL}_{INSFH} (where again $INSFH$ ranges over $FOLEQ$ and HOL) and a specification expression $SP \in SPEX(ASL)$ will return the symbols of the proof system $\Pi^{ASL}_{INSFH}(SP)$ since we won't assume that the symbols of the specification expression SP is equal to the symbols of the proof system $\Pi^{ASL}_{INSFH}(SP)$. We will just assume that there exists an inclusion morphism between $Signature(SP)$ and $Symbols(\Pi^{ASL}_{INSFH}, SP)$. Also, the definition of the consequence relation will use the function Γenv which given a proof system and an specification expression $SP \in SPEX(ASL)$ will return the set of assumptions which we can use for the derivation of sentences from the given

specification expression. Finally, it will also be used the function *Rules* which given a proof system and an specification expression, it will return the proof rules of the proof system.

For any bijective signature morphism $\sigma : Symbols(\Pi^{ASL}_{INSFH}(SP)) \rightarrow \Sigma'$, we will also denote by σ the renaming function which given a proof system $\Pi^{ASL}_{INSFH}(SP)$ will return the resulting proof system of applying to every sentence of every rule of the given proof system the morphism between sentences $\sigma : Sen_{INSFH}\ (Symbols(\Pi^{ASL}_{INSFH}(SP))) \rightarrow Sen_{INSFH}(\Sigma')$.

The definition of the consequence relation associated to these non-compositional proof systems and some of its properties are as follows:

Definition 2.4 *The consequence relation* $\vdash_{\Pi^{ASL}_{INSFH}}$ *which relates specification expressions* $SP \in SPEX(ASL)$ *with sentences* $\phi \in Sen(Symbols(\Pi^{ASL}_{INSFH}, SP))$ *is defined as follows:*

$$SP \vdash_{\Pi^{ASL}_{INSFH}} \phi \Leftrightarrow \textit{Any closed sequent of the form}$$

$$\Gamma env(\Pi^{ASL}_{INSFH}, SP) \Rightarrow_{Symbols(\Pi^{ASL}_{INSFH}, SP), X} \phi \textit{ has a derivation}$$

$$\textit{in } Rules(\Pi^{ASL}_{INSFH}, SP).$$

Definition 2.5 *For any signature* $\Sigma \in AlgSig$, *a* Σ*-algebra* $A \in Alg(\Sigma)$ *satisfies a closed sequent of the form* $\Gamma \Rightarrow_X \phi$ *if the following condition holds:*

$$(\bigwedge_{\psi \in \Gamma} A \models_{\Sigma, INSFH} \psi) \Rightarrow A \models_{\Sigma, INSFH} \phi$$

Definition 2.6 *For any signature* $\Sigma \in AlgSig$, *a* Σ*-algebra* $A \in Alg(\Sigma)$ *satisfies a closed sequent of the form* $X \blacktriangleright \phi : \tau$ *if* $[\![\phi]\!]_{\rho,A} \in [\![\tau]\!]_A$ *for any* $Types_{HOL}(\Sigma)$*-sorted valuation* ρ *such that for any* $x : \tau \in X$, $x \in Dom_\tau(\rho_\tau)$.

Definition 2.7 *For any signature* $\Sigma \in AlgSig$, *a* Σ*-algebra* $A \in Alg(\Sigma)$ *satisfies a closed sequent of the form* $\phi =_{X,\beta} \phi'$ *if* $[\![\phi]\!]_{\rho,A} = [\![\phi']\!]_{\rho,A}$ *for any* $Types_{HOL}(\Sigma)$*-sorted valuation* ρ *such that for any* $x : \tau \in X$, $x \in Dom_\tau(\rho_\tau)$.

Definition 2.8 *For any signature* $\Sigma \in AlgSig$, *a* Σ*-algebra* $A \in Alg(\Sigma)$ *satisfies* Π^{ASL}_{INSFH} *if for all the rules of the natural deduction system the following condition holds:*

- *If A satisfies all the sequents of the premises then A satisfies the sequent of the conclusion.*

Definition 2.9 *A consequence relation* $\vdash_{\Pi^{ASL}_{INSFH}(SP)}$ *is sound if for all specification expressions* $SP \in SPEX(ASL)$ *and for all* $A \in Models(SP)$ *there exists a* $Symbols(\Pi^{ASL}_{INSFH}(SP))$*-algebra* A' *which satisfies the following conditions (which we will refer as soundness conditions):*

- $A'|_{Signature(SP)} = A$.

- $\forall \psi \in \Gamma env(\Pi^{ASL}_{INSFH}, SP).A' \models \psi$.

- A' *satisfies* $Rules(\Pi^{ASL}_{INSFH}, SP)$.

Proposition 2.10 *If* $\phi \in Sen_{AINS}(Signature(SP))$ *and* $\vdash_{\Pi^{ASL}_{INSFH}(SP)}$ *is sound then* $SP \vdash_{\Pi^{ASL}_{INSFH}(SP)} \phi \supset SP \models \phi$.

Proof:
Assume that $SP \vdash_{\Pi^{ASL}_{INSFH}(SP)} \phi$.

We have to show that for all $A \in Models(SP)$ $A \models \phi$.

Let A' be the $Symbols(\Pi^{ASL}_{INSFH}, SP)$-algebra which satisfies the soundness conditions for a given $Signature(SP)$−algebra A.

Since $SP \vdash_{\Pi^{ASL}_{INSFH}(SP)} \phi$, there exists a derivation of the sequent

$$\Gamma env(\Pi^{ASL}_{INSFH}, SP) \Rightarrow_{Symbols(\Pi^{ASL}_{INSFH}, SP), X} \phi$$

and since A' satisfies $\Pi^{ASL}_{INSFH}(SP)$ we can deduce that

$$(\bigwedge_{\psi \in \Gamma env(\Pi^{ASL}_{INSFH}, SP)} A' \models_{Symbols(\Pi^{ASL}_{INSFH}, SP)} \psi) \Rightarrow A' \models_{Symbols(\Pi^{ASL}_{INSFH}, SP)} \phi$$

By the second soundness condition, we have that $A' \models_{Symbols(\Pi^{ASL}_{INSFH}, SP)} \phi$ and since $A'|_{Signature(SP)} = A$ and $\phi \in Sen_{INSFH}(Signature(SP))$, we have that $A \models_{Signature(SP)} \phi$.

Definition 2.11 *A consequence relation* $\vdash_{\Pi^{ASL}_{INSFH}(SP)}$ *is sound and complete if for all specification expressions* $SP \in SPEX(ASL)$, *for all sentences* $\phi \in |Sen_{INSFH}(Symbols(\Pi^{ASL}_{INSFH}, SP))|$ *for all* $A \in Models(SP)$, *if* $A \models \phi$ *there exists a* $Symbols(\Pi^{ASL}_{INSFH}(SP))$*-algebra* A' *such that* A' *satisfies the soundness conditions and* $A' \models \phi$ *if and only if* $SP \vdash_{\Pi^{ASL}_{INSFH}(SP)} \phi$.

Different kind of non-compositional proof systems inductively defined by specification expressions have been developed for the behavioural operators. In [HS96], these operators can just be applied to basic specifications, whereas in [HWB97] they can be applied to any kind of specification expressions. In [HS96] it is developed specific proof systems for the behaviour and the abstract operator for standard and behavioural theories and they use higher-order logic as specification logic, whereas in [HWB97] they present proof systems for the behaviour, abstract and quotient operator but just for standard theories and they use finitary and infinitary first-order logic. In this subsection, we will present in an arbitrary but fixed algebraic institutions, proof systems for the common operators of ASLker specification languages, and after that we will present two proof systems inductively defined by specification expressions for the $BASL$ker

specification language $RBASL$: one with the algebraic institution $FOLEQ$ and the other with the algebraic institution HOL. Again, the design of the proof systems is based on [Hen97] and the axiomatisation of the indistinguishability relation for the proof system of higher-order logic is based on [HS96].

Basically, the proof systems for the behavioural operators are defined as the proof systems of structured specifications whose semantics are equivalent to the semantics of the behavioural operators. The result which states the equivalence is presented in this chapter in theorem 5.2.39 for the proof system of higher-order logic. For first-order logic, the proof system is designed in a similar way but instead of having an axiomatisation of the indistinguishability relation in first-order logic, we have proof rules to derive proofs about indistinguishability.

2.1.1 Proof system for the common operators

Definition 2.12 *Let ASL be any $ASLker$ specification language with an arbitrary but fixed algebraic institution $AINS$. The proof system $\Pi_{AINS}(ASL)$ is inductively defined for the two common operators of $ASLker$ languages as follows:*

$$Rules(\Pi^{ASL}_{AINS}, <\Sigma, \Phi>) = \Pi_{AINS}(\Phi, \Sigma)$$

$$Symbols(\Pi^{ASL}_{AINS}, <\Sigma, \Phi>) = \Sigma$$

$$\Gamma env(\Pi^{ASL}_{AINS}, <\Sigma, \Phi>) = \Phi$$

$$Rules(\Pi^{ASL}_{AINS}, SP_1 +_\Sigma SP_2) = inl''(Rules(\Pi^{ASL}_{AINS}, SP_1)) \cup inr''(Rules(\Pi^{ASL}_{AINS}, SP_2))$$

$$Symbols(\Pi^{ASL}_{AINS}, SP_1 +_\Sigma SP_2) =$$

$$inl''(Symbols(\Pi^{ASL}_{AINS}, SP_1)) \cup inr''(Symbols(\Pi^{ASL}_{AINS}, SP_2))$$

$$\Gamma env(\Pi^{ASL}_{AINS}, SP_1 +_\Sigma SP_2) =$$

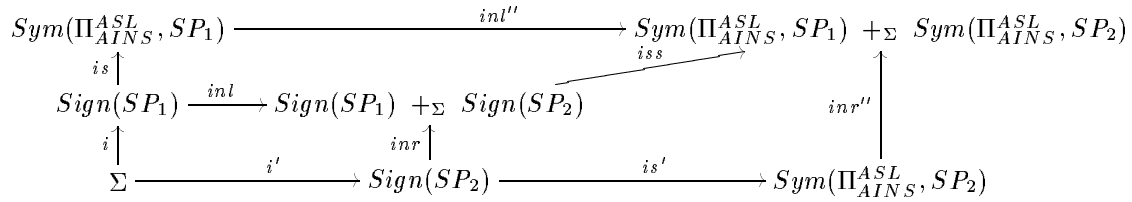
$$inl''(\Gamma env(\Pi^{ASL}_{AINS}, SP_1)) \cup inr''(\Gamma env(\Pi^{ASL}_{AINS}, SP_2))$$

$$Rules(\Pi^{ASL}_{AINS}, SP|_\Sigma) = Rules(\Pi^{ASL}_{AINS}, SP)$$

$$Symbols(\Pi^{ASL}_{AINS}, SP|_\Sigma) = Symbols(\Pi^{ASL}_{AINS}, SP)$$

$$\Gamma env(\Pi^{ASL}_{AINS}, SP|_\Sigma) = \Gamma env(\Pi^{ASL}_{AINS}, SP)$$

where the overloaded symbols inl'' and inr'' are the pushout morphisms of the following diagram:



and inl'' and inr'' also denote the usual renaming functions between environments and proof systems which use the pushouts morphisms with the same name.

Since $Signature(SP_1) +_\Sigma Signature(SP_2)$ is a pushout iss is the unique morphism with arity

$$iss : Signature(SP_1) +_\Sigma Signature(SP_2) \hookrightarrow$$

$$Symbols(\Pi^{ASL}_{AINS}, SP_1) +_\Sigma Symbols(\Pi^{ASL}_{AINS}, SP_2)$$

and the pushouts can be chosen in such a way that iss is an inclusion.

Theorem 2.13 *Let ASLK be the ASLker specification language with just the two common operators of this class of languages. If $\vdash_{\Pi_{AINS}(\emptyset)}$ is sound then $\vdash_{\Pi^{ASLK}_{AINS}}$ is sound and if $\vdash_{\Pi_{AINS}(\emptyset)}$ is sound and complete then $\vdash_{\Pi^{ASLK}_{AINS}}$ is sound and complete.*

2.1.2 The proof system Π^{RBASL}_{FOLEQ}

Definition 2.14 *The natural deduction system $\Pi_{FOLEQ}(\Gamma)$ is defined by the following set of rules for any $\Gamma \in \mathcal{P}(|Sen_{AINS}(\Sigma)|)$:*

$$\frac{}{\{\} \Rightarrow_X \mathbf{true}} \quad (T) \qquad \frac{}{\Gamma \Rightarrow_X \mathbf{false} \supset \phi} \quad (F)$$

$$\frac{\Gamma \Rightarrow_X \phi_1 \wedge \phi_2}{\Gamma \Rightarrow_X \phi_1} \quad (\wedge El) \qquad \frac{\Gamma \Rightarrow_X \phi_1 \wedge \phi_2}{\Gamma \Rightarrow_X \phi_2} \quad (\wedge Er)$$

$$\frac{\Gamma \Rightarrow_X \phi_1 \quad \Gamma \Rightarrow_X \phi_2}{\Gamma \Rightarrow_X \phi_1 \wedge \phi_2} \quad (\wedge I)$$

$$\frac{\Gamma \Rightarrow_X \phi_1}{\Gamma \Rightarrow_X \phi_1 \vee \phi_2} \quad (\vee Il) \qquad \frac{\Gamma \Rightarrow_X \phi_2}{\Gamma \Rightarrow_X \phi_1 \vee \phi_2} \quad (\vee Ir)$$

$$\frac{\Gamma \Rightarrow_X \phi_1 \vee \phi_2 \quad \Gamma \Rightarrow_X \phi_1 \supset \psi \quad \Gamma \Rightarrow_X \phi_2 \supset \psi}{\Gamma \Rightarrow_X \psi} \quad (\vee E)$$

$$\frac{\Gamma \cup \{\phi\} \Rightarrow_X \mathbf{false}}{\Gamma \Rightarrow_X \neg\phi} \quad (\neg I) \qquad \frac{\Gamma \Rightarrow_X \psi \quad \Gamma \Rightarrow_X \neg\psi}{\Gamma \Rightarrow_X \phi} \quad (\neg E)$$

$$\frac{\Gamma \cup \phi \Rightarrow_X \phi'}{\Gamma \Rightarrow_X \phi \supset \phi'} \quad (\supset\ i) \qquad \frac{\Gamma \Rightarrow_X \phi \supset \phi' \quad \Gamma \Rightarrow_X \phi}{\Gamma \Rightarrow_X \phi'} \quad (\supset\ e)$$

$$\frac{\Gamma \Rightarrow_X \phi[t/x]}{\Gamma \Rightarrow_X \exists x : s.\phi}\ t \in T_{\Sigma,s}(X) \qquad (\exists I)$$

$$\frac{\Gamma \Rightarrow_X \exists x : s.\phi \quad \Gamma \cup \{\phi\} \Rightarrow_{X \cup \{x:s\}} \psi}{\Gamma \Rightarrow_X \psi} \qquad (\exists E)$$

$$\frac{\Gamma \Rightarrow_{X \cup \{x:s\}} \phi}{\Gamma \Rightarrow_X \forall x : s.\phi} \qquad (\forall\ I)$$

$$\frac{\Gamma \Rightarrow_X \forall x : s.\phi}{\Gamma \Rightarrow_X \phi\{t/x\}}\ t \in T_{\Sigma,s}(X) \qquad (\forall\ E)$$

$$\frac{}{\Gamma \Rightarrow_X t = t} \quad (REFL) \qquad \frac{\Gamma \Rightarrow_X r = s \quad \Gamma \Rightarrow_X s = t}{\Gamma \Rightarrow_X r = t} \qquad (TRANS)$$

$$\frac{\Gamma \Rightarrow_X t = s}{\Gamma \Rightarrow_X s = t} \quad (SYM) \qquad \frac{\Gamma \Rightarrow_{X \cup \{x:s'\}} \phi[s/x] \quad \Gamma \Rightarrow_X t = s}{\Gamma \Rightarrow_{X \cup \{x:s'\}} \phi[t/x]} \qquad (SUBST)$$

In the next definitions, we present different relational signatures which extend a given signature with the following symbols:

- Symbols to denote the indistinguishability relation for every sort of the signature and symbols to denote a definedness predicate also for every sort of the signature.
- The same extension as in the previous one plus symbols to denote a pseudoepimorphism between the original signature and a disjoint copy.
- The same extension as in the previous one plus symbols to denote contexts and context application.

Definition 2.15 *The relational signature* $\Sigma[\sim, D]$ *is defined for any signature* $\Sigma = (S, Op)$ *and for any S-families of new symbols* $\sim, D$ *as:*

$$\Sigma[\sim, D] = \Sigma \cup \{\sim_s : s \times s | s \in S\} \cup \{D_s : s | s \in S\}$$

Definition 2.16 *The relational signature* $\Sigma[\sim, D, \pi_{Copy}]$ *for any signature* $\Sigma = (S, Op)$*, for any bijective signature morphism* $Copy : \Sigma \to Copy(\Sigma)$ *such that* $\Sigma \cap Copy(\Sigma) = \emptyset$ *and for any* $S-$*family of new symbols* $\sim, D$ *and* π *is defined as:*

$$\Sigma[\sim, D, \pi_{Copy}] = \Sigma[\sim, D] \cup Copy(\Sigma) \cup \{\pi_s : s \to Copy(s) | s \in S\}$$

Remark: *The relational signature* $Copy(\Sigma)[\sim, D, \pi_{Copy^{-1}}]$ *stands for the following signature:*

$$Copy(\Sigma)[\sim, D, \pi_{Copy^{-1}}] = Copy(\Sigma)[\sim, D] \cup \Sigma \cup \{\pi_s : Copy(s) \to s | s \in S\}$$

Definition 2.17 *The relational signature* $\Sigma[\sim, D, \pi_{Copy}, z, c, c\_appl]$ *for any signature* $\Sigma = (S, Op)$*, for any bijective signature morphism* $Copy : \Sigma \to Copy(\Sigma)$ *such that* $\Sigma \cap Copy(\Sigma) = \emptyset$*, for any* $S-$*family of new symbols* $\sim$*,* π *and* z *and for any new symbols* c*,* c_appl *is defined as follows:*

$$Sorts(\Sigma[\sim, D, \pi_{Copy}, z, c, c\_appl]) \;=$$

$$Sorts(\Sigma)[\sim, D, \pi_{Copy}] \;\cup\; \{c[r \;\to\; s] \mid r \in S, s \in S\}$$

$$Ops(\Sigma[\sim, D, \pi_{Copy}, z, c, c\_appl]) \;=\; Ops(\Sigma[\sim, D, \pi_{Copy}]) \;\cup$$

$$\{c[z_s] \;:\; c[s \;\to\; s] \mid s \in S\} \;\cup$$

$$\{c[r, f] : c[r \;\to\; s_1] \times \ldots \times c[r \;\to\; s_n] \;\to\; c[r \;\to\; s] \mid$$

$$r \in S, f : s_1 \times \ldots \times s_n \to s \in Op\} \;\cup$$

$$\{c\_appl[r, s] : c[r \;\to\; s] \;\times\; r \;\to\; s \mid r \in S, s \in S\}$$

$$Prs(\Sigma[\sim, D, \pi_{Copy}, z, c, c\_appl]) \;=\; Pr(\Sigma[\sim, D])$$

In the following, some proof rules to prove sentences of the form $\forall x : S.D_s(x) \supset \phi$ are presented:

Definition 2.18 *The set of rules* $D\_sr[\Sigma[\sim, D], In]$ *is defined as follows:*

$$D\_sr[\Sigma[\sim, D], In] \;=\; \{D\_sr[\Sigma[\sim, D], In, s] \mid s \in In\}$$

and the rule $D\_sr[\Sigma[\sim, D], In, s]$ *is defined for every sort* $s \in S - In$ *as follows:*

$$\frac{\{\Gamma \Rightarrow_X \forall x_1 : s_1. \ldots . \forall x_n : s_n. \bigwedge_{s_i = s} \phi\{x_i \,/\, x\} \;\supset\; \phi\{f(x_1, \ldots, x_n) \,/\, x\} \mid f : s_1 \;\times\; \ldots \;\times\; s_n \;\to\; s \in Ops(\Sigma)\}}{\Gamma \Rightarrow_X \forall x : s.D_s(x) \supset \phi}$$

Next, proof rules to define the proof system associated to the reachability operator are presented:

Definition 2.19 *The set of rules* $Reach\_sr[\Sigma, (\mathcal{S}_\mathcal{R}, \mathcal{F}_\mathcal{R})]$ *is defined as follows:*

$$Reach\_sr[\Sigma, (\mathcal{S}_\mathcal{R}, \mathcal{F}_\mathcal{R})] \;=\; \{Reach\_sr[\Sigma, (\mathcal{S}_\mathcal{R}, \mathcal{F}_\mathcal{R}), s] \mid s \in \mathcal{S}_\mathcal{R}\}$$

and the rule $Reach\_sr[\Sigma, (\mathcal{S}_{\mathcal{R}}, \mathcal{F}_{\mathcal{R}}), s]$ *is defined for every sort* $s \in \mathcal{S}_{\mathcal{R}}$ *as follows:*

$$\frac{\{\Gamma \Rightarrow_X \forall x_1 : s_1. \ldots . \forall x_n : s_n. \bigwedge_{s_i = s} \phi\{x_i \,/\, x\} \supset \phi\{f(x_1, \ldots, x_n) \,/\, x\} \mid f : s_1 \times \ldots \times s_n \to s \in \mathcal{F}_{\mathcal{R}}\}}{\Gamma \Rightarrow_X \forall x : s.\phi}$$

In the following, proof rules to derive proofs about indistinguishability are presented. The proof rules appear in the proof system of the behavioural operators and there are two different kind of proof rules:

- proof rules which define the indistinguishability relation.
- proof rules to perform context induction to reason about formulas of the form $\forall cxt : c[r \to s].\phi$

Definition 2.20 *The set of rules* $Indist\_rel\_sr[\Sigma[\sim]]$ *is defined as follows:*

- *For each sort* $s \in S - Obs$*, the following rule:*

$$\frac{\Gamma \Rightarrow_X \bigwedge_{obs \in Obs} \forall cxt : c[r \to obs]. \; c\_appl[s, obs](cxt, t) \;=\; c\_appl[s, obs](cxt, r)}{\Gamma \Rightarrow_X t \sim_s r} \; D_s(t) \& D_s(r)$$

- *The following context induction rule for each* $r \in S$ *and* $s \in S - In$ *such that* $r \neq s$*:*

$$\frac{\{\Gamma \Rightarrow_X \forall cxt_1 : c[r \to s_1]. \ldots . \forall cxt_n : c[r \to s_n]. \bigwedge_{s_i = s} \phi\{cxt_i/cxt\} \supset \phi\{c[r, f](cxt_1, \ldots, cxt_n)/cxt\} \mid f : s_1 \times \ldots \times s_n \to s \in Op\}}{\Gamma \Rightarrow_X \forall cxt : c[r \to s].\phi}$$

- *The following context induction rule for each* $r \in S$ *and* $s \in S - In$ *such that* $r = s$*:*

$$\frac{\{\Gamma \Rightarrow_X \forall cxt_1 : c[r \to s_1]. \ldots . \forall cxt_n : c[r \to s_n]. \phi\{c[z_s]/cxt\} \wedge \bigwedge_{s_i = s} \phi\{cxt_i/cxt\} \supset \phi\{c[r, f](cxt_1, \ldots, cxt_n)/cxt\} \mid f : s_1 \times \ldots \times s_n \to s \in Op\}}{\Gamma \Rightarrow_X \forall cxt : c[r \to s].\phi}$$

- *For each sort* $s \in Obs$*, the following rule:*

$$\frac{\Gamma \vdash_X t = r}{\Gamma \vdash_X t \sim_s r} \; D_s(t) \& D_s(r)$$

In the following, you will find the axiomatisation of the pseudoepimorphism used in the proof system for the behavioural operator. This axiomatisation is formulated over the signatur $\Sigma[\sim, D, \pi_{Copy}]$ and it contains the following axioms:

- axioms to determine that π_{Copy} is an homomorphism.
- axioms to determine that the homomorphism π_{Copy} is surjective.
- axioms to establish the compatibility between the indistinguishability relation for every sort of the signature and the set theoretical equality associated to the disjoint copy of the given sort.

Definition 2.21 *The set of sentences* $pEpi_{FOLEQ}[\Sigma[\sim, D, \pi_{Copy}]]$ *is defined as:*

$$pEpi_{FOLEQ}[\Sigma[\sim, D, \pi_{Copy}], Obs, In] = Hom[\Sigma[\sim, D, \pi_{Copy}], Obs, In] \cup$$

$$Epihom[\Sigma[\sim, D, \pi_{Copy}], Obs, In] \cup \sim\text{-}comp[\Sigma[\sim, D, \pi_{Copy}], Obs, In]$$

where

$$Hom[\Sigma[\sim, \pi_{Copy}], Obs, In] = \bigcup_{f:s_1\times\ldots\times s_n\to s\in Op} \{\forall t_1 : s_1 \ldots . \forall t_n : s_n .$$

$$\bigwedge_{i\in[1..n], s_i\in S-In} D_{s_i}(t_i) \Rightarrow$$

$$\pi_{Copy,s}(f(t_1, \ldots, t_n)) = Copy(f)(\pi_{Copy,s_1}(t_1), \ldots, \pi_{Copy,s_n}(t_n))\}$$

$$Epihom[\Sigma[\sim, \pi_{Copy}], Obs, In] =$$

$$\bigcup_{s\in S-In} \{\forall y : Copy(s).\exists x : s. D_s[\Sigma, In](x) \wedge \pi_{Copy,s}(x) = y)\}$$

$$\bigcup_{s\in In} \{\forall y : Copy(s).\exists x : s.\ \pi_{Copy,s}(x) = y)\}$$

$$\sim\text{-}comp[\Sigma[\sim, \pi_{Copy}], Obs, In] =$$

$$\bigcup_{s\in S-In} \{\forall x : s.\forall y : s. D_s[\Sigma, In](x) \wedge D_s[\Sigma, In](y) \Rightarrow$$

$$x \sim_s y \Leftrightarrow \pi_{Copy,s}(x) = \pi_{Copy,s}(y)\}$$

$$\bigcup_{s\in In} \{\forall x : s.\forall y : s. x \sim_s y \Leftrightarrow \pi_{Copy,s}(x) = \pi_{Copy,s}(y)\}$$

Finally, we present an axiomatisation of the function application between contexts and values.

Definition 2.22 *The set of sentences* $Ax\_c\_appl[\Sigma[\sim, D, \pi_{Copy}, z, c, c\_appl]]$ *is defined as:*

$$Ax\_c\_appl[\Sigma[\sim, D, \pi_{Copy}, z, c, c\_appl]] \ =$$

$$\{Ax\_c\_appl[\Sigma[\sim, D, \pi_{Copy}, z, c, c\_appl], r, s] \mid r \in S, s \in S\}$$

where

$$Ax\_c\_appl[\Sigma[\sim, D, \pi_{Copy}, z, c, c\_appl], r, s] \ =$$

$$\begin{cases} \forall v : r. c\_appl[r,s](c[z_r], v) \ = \ v \wedge c\_appl[r,s](x,v) \ = \ x \wedge \\ \bigwedge_{f:s_1 \times \ldots \times s_n \to s \in Op} \forall cxt_1 : c[r \to s_1]. \ldots . \forall cxt_n : c[r \to s_n]. \forall v : r. \\ \quad c\_appl[r,s](c[r,f](cxt_1, \ldots, cxt_n), v) = \\ \quad\quad f(c\_appl[r,s_1](cxt_1, v), \ldots, c\_appl[r,s_n](cxt_n, v)) \qquad , if \ r = s \\ \\ \forall v : r. \, c\_appl[r,s](x,v) \ = \ x \wedge \\ \bigwedge_{f:s_1 \times \ldots \times s_n \to s \in Op} \forall cxt_1 : c[r \to s_1]. \ldots . \forall cxt_n : c[r \to s_n]. \forall v : r. \\ \quad c\_appl[r,s](c[r,f](cxt_1, \ldots, cxt_n), v) = \\ \quad\quad f(c\_appl[r,s_1](cxt_1, v), \ldots, c\_appl[r,s_n](cxt_n, v)) \qquad , if \ r \neq s \end{cases}$$

Definition 2.23 *The proof system* Π^{RBASL}_{FOLEQ} *is inductively defined for the specific operators of the language as follows:*

$$Rules(\Pi^{RBASL}_{FOLEQ}, rename \quad SP \quad \textbf{by} \quad \sigma) = \sigma''(Rules(\Pi^{RBASL}_{FOLEQ}, SP))$$

$$Symbols(\Pi^{RBASL}_{FOLEQ}, rename \quad SP \quad \textbf{by} \quad \sigma) =$$

$$\sigma''(Symbols(\Pi^{RBASL}_{FOLEQ}, SP))$$

$$\Gamma env(\Pi^{RBASL}_{FOLEQ}, rename \quad SP \quad \textbf{by} \quad \sigma) =$$

$$\sigma''(\Gamma env(\Pi^{RBASL}_{FOLEQ}, SP))$$

$$Rules(\Pi^{RBASL}_{FOLEQ}, \textbf{reach} \quad SP \quad \textbf{with} \quad (\mathcal{S}_{\mathcal{R}}, \mathcal{F}_{\mathcal{R}})) \; = \; Rules(\Pi^{RBASL}_{FOLEQ}, SP) \cup$$

$$Reach\_sr[Signature(SP), (\mathcal{S}_{\mathcal{R}}, \mathcal{F}_{\mathcal{R}})]$$

$$Symbols(\Pi^{RBASL}_{FOLEQ}, \textbf{reach} \quad SP \quad \textbf{with} \quad (\mathcal{S}_{\mathcal{R}}, \mathcal{F}_{\mathcal{R}})) \; =$$

$$Symbols(\Pi^{RBASL}_{FOLEQ}, SP)$$

$$\Gamma env(\Pi^{RBASL}_{FOLEQ}, \textbf{reach} \quad SP \quad \textbf{with} \quad (\mathcal{S}_{\mathcal{R}}, \mathcal{F}_{\mathcal{R}})) \; = \; \Gamma env(\Pi^{RBASL}_{FOLEQ}, SP)$$

$$Rules(\Pi^{RBASL}_{FOLEQ}, \textbf{behaviour} \quad SP \quad \textbf{wrt} \quad \approx) =$$

$$Copy''(Rules(\Pi^{RBASL}_{FOLEQ}, SP)) \cup$$

$$Indist\_rel\_sr[\Sigma[\sim], Obs, In] \; \cup \; D\_sr[\Sigma[\sim, D], In]$$

$$Symbols(\Pi^{RBASL}_{FOLEQ}, \textbf{behaviour} \quad SP \quad \textbf{wrt} \quad \approx) =$$

$$Copy''(Symbols(\Pi^{RBASL}_{FOLEQ}, SP)) \cup \Sigma[\sim, \pi_{Copy}, z, c, c\_appl]$$

$$\Gamma env(\Pi^{RBASL}_{FOLEQ}, \textbf{behaviour} \quad SP \quad \textbf{wrt} \quad \approx) =$$

$$\Gamma env(rename \quad SP \quad \textbf{by} \quad Copy) \cup$$

$$pEpi_{FOLEQ}[\Sigma[\sim, \pi_{Copy}, z, c, c\_appl]] \; \cup$$

$$Ax\_c\_appl[\Sigma[\sim, \pi_{Copy}, z, c, c\_appl]] >)$$

$$Rules(\Pi^{RBASL}_{FOLEQ}, \textbf{abstract} \quad SP \quad \textbf{by} \quad \equiv) =$$

$$Rules(\Pi^{RBASL}_{FOLEQ}, \textbf{behaviour} \quad SP/\approx \quad \textbf{wrt} \quad \approx)$$

$$Symbols(\Pi^{RBASL}_{FOLEQ}, \textbf{abstract} \quad SP \quad \textbf{by} \quad \equiv) =$$

$$Symbols(\Pi^{RBASL}_{FOLEQ}, \textbf{behaviour} \quad SP/\approx \quad \textbf{wrt} \quad \approx)$$

$$\Gamma env(\Pi^{RBASL}_{FOLEQ}, \textbf{abstract} \quad SP \quad \textbf{by} \quad \equiv) =$$

$$\Gamma env(\Pi^{RBASL}_{FOLEQ}, \textbf{behaviour} \quad SP/\approx \quad \textbf{wrt} \quad \approx)$$

$$Rules(\Pi^{RBASL}_{FOLEQ}, SP/\approx) =$$

$$Rules(\Pi^{RBASL}_{FOLEQ}, rename \quad SP \quad \textbf{by} \quad Copy) \cup$$

$$Indist\_rel\_sr[Copy(\Sigma)[\sim, \pi_{Copy-1}, copyz, copyc, copyc\_appl],$$

$$Copy(Obs), Copy(In)] \ \cup \ D\_sr[Copy(\Sigma)[\sim, D], Copy(In)]$$

$$Symbols(\Pi^{RBASL}_{FOLEQ}, SP/\approx) =$$

$$Copy''(Symbols(SP, \Pi^{RBASL}_{FOLEQ})) \cup$$

$$Copy(\Sigma)[\sim, \pi_{Copy-1}, copyz, copyc, copyc\_appl]$$

$$\Gamma env(\Pi^{RBASL}_{FOLEQ}, SP/\approx) =$$

$$\Gamma env(rename \quad SP \quad \textbf{by} \quad Copy) \cup$$

$$pEpi_{FOLEQ}[Copy(\Sigma)[\sim, \pi_{Copy-1}, copyz, copyc, copyc\_appl],$$

$$Copy(Obs), Copy(In)]] \ \cup$$

$$Ax\_c\_appl[Copy(\Sigma)[\sim, \pi_{Copy-1}, copyz, copyc, copyc\_appl],$$

$$Copy(Obs), Copy(In)]] >$$

where Σ is the signature of the argument specification SP for every case and the overloaded function symbol σ'' is used here as the pushout of the following diagram:

$$\begin{array}{ccc} Sym(\Pi^{RBASL}_{FOLEQ}, SP) & \xrightarrow{\sigma''} & PO(i, \sigma'') \\ {\scriptstyle i}\uparrow & & \uparrow \\ \Sigma & \xrightarrow{\quad\sigma\quad} & \Sigma' \end{array}$$

where $i : \Sigma \hookrightarrow Sym(\Pi^{RBASL}_{FOLEQ}, SP)$ and it is also used as the renaming function of environments and of the proof system $\Pi^{RBASL}_{FOLEQ}(SP)$ which use the pushout just described in the obvious and standard way, and the overloaded symbol $Copy''$ is the pushout morphism of the following diagram:

$$\begin{array}{ccc} Sym(\Pi^{RBASL}_{FOLEQ}, SP) & \xrightarrow{Copy''} & PO(i, Copy) \\ {\scriptstyle i}\uparrow & & \uparrow \\ \Sigma & \xrightarrow{\quad Copy\quad} & Copy(\Sigma) \end{array}$$

where $i : \Sigma \hookrightarrow Symbols(\Pi^{RBASL}_{FOLEQ}, SP)$ *and*

$$Copy''\ (Symbols\ (\Pi^{RBASL}_{FOLEQ}, SP)) \cap$$

$$\Sigma[\sim, \pi_{Copy}, z, c, c\_appl] = \ Copy(\Sigma)$$

and also

$$Copy''\ (Symbols(\Pi^{RBASL}_{FOLEQ}, SP))\ \ \cap\ Copy(\Sigma)$$

$$[\sim, \pi_{Copy-1}, copyz, copyc, copyc\_appl]\ =\ Copy(\Sigma)$$

The symbol $Copy''$ *is also used as the renaming functions of environmnents and of set of rules using the pushout morphism just described with the same name in the obvious and standard way.*

Note that although the proof system for the behaviour and quotient operators is defined in terms of the proof system of a structured specification expression, it is not needed to apply to the symbols of these proof systems the pushouts to avoid name clashes in structured specifications since the conditions which satisfy the pushout $Copy''$ *guarantee no name clashes.*

Theorem 2.24 *The consequence relation* $\vdash_{\Pi^{RBASL}_{FOLEQ}}$ *is sound.*

Proof:

The proof is by induction on specification expressions. For the two common operators the proof is trivial since $\vdash_{\Pi_{FOLEQ}(\emptyset)}$ is sound. For the rest of the cases, we have to prove the following propositions:

- For all specification expressions $SP \in SPEX(RBASL)$ such that $Signature(SP) = \Sigma = (S, Op)$, if $\vdash_{\Pi^{RBASL}_{FOLEQ}(SP)}$ is sound then $\vdash_{\Pi^{RBASL}_{FOLEQ}(SP|_\Sigma)}$ is sound which follows trivially.

- For all specification expressions $SP \in SPEX(RBASL)$ such that $Signature(SP) = \Sigma = (S, Op)$, if $\vdash_{\Pi^{RBASL}_{FOLEQ}(SP)}$ is sound then $\vdash_{\Pi^{RBASL}_{FOLEQ}(rename\quad SP\quad \mathbf{by}\quad \sigma)}$ is sound where $\sigma : Signature(SP) \rightarrow \Sigma'$.
  Assume that $SP \in SPEX(RBASL)$ and assume that $\vdash_{\Pi^{RBASL}_{FOLEQ}(SP)}$ is sound.
  We have to prove that for any $A \in Models(rename\quad SP\quad \mathbf{by}\quad \sigma)$ there exists a $Symbols(\Pi^{RBASL}_{FOLEQ}(rename\quad SP\quad \mathbf{by}\quad \sigma))$-algebra (which we will refer as A') such that:

  - $A'|_{\Sigma'} = A$.

  - $\forall \psi \in \Gamma env(\Pi^{ASL}_{AINS}, rename\quad SP\quad \mathbf{by}\quad \sigma).A' \models \psi$.

  - A' satisfies $\Pi^{ASL}_{AINS}(rename\quad SP\quad \mathbf{by}\quad \sigma)$.

Since $A|_\sigma \in Models(SP)$, by soundness of $\vdash_{\Pi^{RBASL}_{FOLEQ}(SP)}$, we know that there exists a $Symbols(\Pi^{RBASL}_{FOLEQ}(SP)$-algebra which we will refer as A'' such that:

- $A''|_\Sigma = A|_\sigma$.
- $\forall \psi \in \Gamma env(\Pi^{ASL}_{AINS}, SP).A' \models \psi$.
- A'' satisfies $\Pi^{ASL}_{AINS}(SP)$.

It is straightforward to prove that the $Symbols(\Pi^{RBASL}_{FOLEQ}$ $(rename$ SP **by** $\sigma))$ -algebra $A' = A''|_{(i;\sigma'')-1}$ where σ'' is the pushout morphism of $i : \Sigma \hookrightarrow Symbols(\Pi^{ASL}_{AINS}(SP))$ and $\sigma : \Sigma \to \Sigma'$ of the definition the proof system for this operator, satisfies the soundness conditions of the satisfaction condition of $AINS$ and because A'' satisfies the soundness conditions for the proof system $\Pi^{ASL}_{AINS}(SP)$.

- For all specification expressions $SP \in SPEX(RBASL)$ such that $Signature(SP) = \Sigma = (S, Op)$, if $\vdash_{\Pi^{RBASL}_{FOLEQ}(SP)}$ is sound then $\vdash_{\Pi^{RBASL}_{FOLEQ}(\textbf{reach}\ \ SP\ \ \textbf{with}\ \ (\mathcal{S}_\mathcal{R}, \mathcal{F}_\mathcal{R}))}$ is sound.
  Assume that $SP \in SPEX(RBASL)$ and assume that $\vdash_{\Pi^{RBASL}_{FOLEQ}(SP)}$ is sound.
  To prove that $\vdash_{\Pi^{RBASL}_{FOLEQ}(\textbf{reach}\ \ SP\ \ \textbf{with}\ \ (\mathcal{S}_\mathcal{R}, \mathcal{F}_\mathcal{R}))}$ is sound, we have to define for any $A \in Models(\textbf{reach}\ \ SP\ \ \textbf{with}\ \ (\mathcal{S}_\mathcal{R}, \mathcal{F}_\mathcal{R}))$,
  a $Symbols(\Pi^{RBASL}_{FOLEQ}(\textbf{reach}\ \ SP\ \ \textbf{with}\ \ (\mathcal{S}_\mathcal{R}, \mathcal{F}_\mathcal{R})))$-algebra B which satisfies the soundness conditions.

  Assume that $A \in Models(\textbf{reach}\ \ SP\ \ \textbf{with}\ \ (\mathcal{S}_\mathcal{R}, \mathcal{F}_\mathcal{R}))$. By the semantics of the reachability operator, we know also that $A \in Models(SP)$.

  Since $\Pi^{RBASL}_{FOLEQ}(SP)$ is sound, we know that there exists a $Symbols(\Pi^{RBASL}_{FOLEQ}(SP))$-algebra A' such that A' satisfies the soundness conditions and therefore $A'|_{Signature(SP)} = A$.

  Assume that B is a $Symbols(\Pi^{RBASL}_{FOLEQ}(SP))$-algebra which satisfies the soundness conditions and $B'|_{Signature(SP)} = A$. To prove that B satisfies the soundness conditions we have to show that:

  - $B|_{Signature(SP)} =$ A which obviously holds
  - $\forall \psi \in \Gamma env(\Pi^{ASL}_{AINS}, \textbf{reach}\ \ SP\ \ \textbf{with}\ \ (\mathcal{S}_\mathcal{R}, \mathcal{F}_\mathcal{R})).B \models \psi$ which holds since $\Gamma env(\Pi^{ASL}_{AINS}, \textbf{reach}\ \ SP\ \ \textbf{with}\ \ (\mathcal{S}_\mathcal{R}, \mathcal{F}_\mathcal{R})) = \Gamma env(\Pi^{ASL}_{AINS}, SP)$.
  - B satisfies $\Pi^{RBASL}_{FOLEQ}(\textbf{reach}\ \ SP\ \ \textbf{with}\ \ (\mathcal{S}_\mathcal{R}, \mathcal{F}_\mathcal{R}))$ because B satisfies
    $\Pi^{RBASL}_{FOLEQ}(SP)$ and for each sort $s \in \mathcal{S}_\mathcal{R}$ B satisfies $Reach\_sr[\Sigma, (\mathcal{S}_\mathcal{R}, \mathcal{F}_\mathcal{R}), s]$

and it follows trivially by term induction on the carrier set B_s since $B|_{Signature(SP)} = A$ satisfies the reachability constraint.

- For all specification expressions $SP \in SPEX(RBASL)$ such that $Signature(SP) = \Sigma = (S, Op)$, if $\vdash_{\Pi^{RBASL}_{FOLEQ}(SP)}$ is sound then $\vdash_{\Pi^{RBASL}_{FOLEQ}(\textbf{behaviour} \quad SP \quad \textbf{wrt} \quad \approx)}$ is sound.
  Assume that $SP \in SPEX(RBASL)$ and assume that $\vdash_{\Pi^{RBASL}_{FOLEQ}(SP)}$ is sound.
  To prove that $\vdash_{\Pi^{RBASL}_{FOLEQ}(\textbf{behaviour} \quad SP \quad \textbf{wrt} \quad \approx)}$ is sound, we have to define for any $A \in Models(\textbf{behaviour} \quad SP \quad \textbf{wrt} \quad \approx)$,
  a $Symbols(\Pi^{RBASL}_{FOLEQ}(\textbf{behaviour} \quad SP \quad \textbf{wrt} \quad \approx))$-algebra B such that:
  - $B|_{Signature(SP)} = \mathrm{A}$
  - $\forall \psi \in \Gamma env(\Pi^{ASL}_{AINS}, SP).B \models \psi$.
  - B satisfies $\Pi^{RBASL}_{FOLEQ}(\textbf{behaviour} \quad SP \quad \textbf{wrt} \quad \approx)$

  Since $\Pi^{RBASL}_{FOLEQ}(SP)$ is sound, there exists a $Symbols(\Pi^{RBASL}_{FOLEQ}(SP))$-algebra A' such that $A'|_{Signature(SP)} = A/{\approx}^{In,Obs}_{A}$ and A' satisfies $\Pi^{RBASL}_{FOLEQ}(SP)$.
  The $Symbols(\Pi^{RBASL}_{FOLEQ}(\textbf{behaviour} \quad SP \quad \textbf{wrt} \quad \approx))$-algebra B is defined as follows:

  $B|_{Signature(SP)} = \mathrm{A}$ (and therefore it satisfies the first condition).

  $B|_{Copy''(Symbols(\Pi^{RBASL}_{FOLEQ}(SP)))} = A'|_{Copy''-1}$

  $B|_{\pi_s} = \epsilon_{A,s}$ for every sort $s \in S$.

  where

  $\epsilon_{A,s}(v) = [v]_{\approx^{Obs,In}_{A,s}}$ for all $v \in A_s$.

$B|_{\sim_s} = \approx_{A,s}^{In,Obs}$ for every sort $s \in S$.

$B|_{c[r,s]} = \{\lambda frv : A[In]_r.I_{\alpha \cup \{(z_r, frv)\}}(czr) \mid$

$\alpha \in (X_{In} \to A)$, $czr \in T_{\Sigma,s}(X_{In} \cup \{z_r\})\}$for every sort $s, r \in S$

and for a S-sorted infinite enumerable set of variables such that

for every sort $s \in S$ $X_s \cap z_s = \emptyset$.

$c[z_s]_B = z_s$ for every sort $s \in S$

$c[r,f]_B(cx_1, \ldots, cx_n) = f(cx_1, \ldots, cx_n)$ for every sort $r \in S$,

$f : s_1 \times \ldots \times s_n \to s \in Op$,

$cx_1 \in T_{\Sigma,s_1}(X_{In} \cup \{z_r\})$, ... , $cx_n \in T_{\Sigma,s_n}(X_{In} \cup \{z_r\})$

$c\_appl[r,s]_B(\lambda frv : A[In]_r.I_{\alpha \cup \{(z_r, frv)\}}(czr), rv) =$

$I_{\alpha \cup \{(z_r, rv)\}}(czr)$ for all $\alpha : X_{In} \to A[In]$,

for all $czr \in T_{\Sigma,s}(X_{In} \cup \{z_r\})$ and for all $rv \in A[In]_s$

To show that

$$\forall \psi \in \Gamma env(\Pi_{AINS}^{ASL}, \textbf{behaviour} \quad SP \quad \textbf{wrt} \quad \approx).B \models \psi$$

we have to prove the following propositions:

- $\forall \psi \in \Gamma env(\Pi_{AINS}^{ASL}, rename \quad SP \quad \textbf{by} \quad Copy).B \models \psi$ which follows by the induction hypotheses and the proof of soundness for the rename operator.
- $B \models pEpi_{FOLEQ}[\Sigma[\sim, \pi_{Copy}, z, c, c\_appl]] \wedge$

  $B \models Ax\_c\_appl[\Sigma[\sim, \pi_{Copy}, z, c, c\_appl]]$

  $B \models pEpi_{FOLEQ}[\Sigma[\sim, \pi_{Copy}, z, c, c\_appl]]$ because for every sort s B_{π_s} can be seen as an epimorphism between A and $A/\approx_{A,s}^{Obs,In}$ and

therefore B satisfies $Hom[\Sigma[\pi_{Copy}],\ Epihom[\Sigma[\pi_{Copy}]]$ and also $\sim -comp[\Sigma[\sim, \pi_{Copy}]]$. The proof of $B \models Ax\_c\_appl[\Sigma[\sim, \pi_{Copy}, z, c, c\_appl]]$ is straightforward.

And to show that B satisfies $\Pi^{RBASL}_{FOLEQ}(\textbf{behaviour}\quad SP\quad \textbf{wrt}\quad \approx)$ we have to show the following propositions:

- $B|_{Copy''(Symbols(\Pi^{RBASL}_{FOLEQ}(SP)))}$ satisfies $\Pi^{RBASL}_{FOLEQ}(rename\quad SP\quad \textbf{by}\quad Copy)$.
  Since $A \in Models(\textbf{behaviour}\quad SP\quad \textbf{wrt}\quad \approx)$, we have that $A/\approx^{Obs,In}_{A} \in Models\ (SP)$ and by induction hypotheses we have that A' satisfies $\Pi^{RBASL}_{FOLEQ}(SP)$ where $A'|_{Signature(SP)} = A/\approx^{Obs,In}_{A}$. Therefore, $B|_{Copy''(Symbols(\Pi^{RBASL}_{FOLEQ}(SP)))}$ satisfies $\Pi^{RBASL}_{FOLEQ}\ (rename\quad SP\quad \textbf{by}\quad Copy)$.
- B satisfies $Indist\_rel\_sr[\Sigma[\sim, \pi_{Copy}, z, c, c\_appl], Obs, In]$.
  B satisfies

$$\frac{\Gamma \Rightarrow_X \bigwedge_{obs \in Obs} \forall cxt : c[r \to obs].\ c\_appl[r, obs](cxt, t) \ =\ c\_appl[r, obs](cxt, r)}{\Gamma \Rightarrow_X t \sim_s r}\ D_s\big[\Sigma, In\big](t) \& D_s\big[\Sigma, In\big](r)$$

  for each sort $s \in S - Obs$ and

$$\frac{\Gamma \vdash_X t = r}{\Gamma \vdash_X t \sim_s r}\ D_s\big[\Sigma, In\big](t) \& D_s\big[\Sigma, In\big](r)$$

  for each sort $s \in S$ because $B|_{\sim} = \approx^{In,Obs}_{A,s}$.
  B satisfies

$$\frac{\{\Gamma \Rightarrow_X \forall cxt_1 : c[r \to s_1]. \ldots . \forall cxt_n : c[r \to s_n].\ \bigwedge_{s_i = s} \phi\,\{cxt_i/cxt\} \ \supset \phi\,\{c[r, f](cxt_1, \ldots, cxt_n)/cxt\} \mid f : s_1 \times \ldots \times s_n \to s \in Op\}}{\Gamma \Rightarrow_X \forall cxt : c[r \to s].\phi}$$

  by induction on the context $cxt : c[r \to s]$

- For all specification expressions $SP \in SPEX(RBASL)$ such that $Signature(SP) = \Sigma = (S, Op)$, if $\vdash_{\Pi^{RBASL}_{FOLEQ}(SP)}$ is sound then $\vdash_{\Pi^{RBASL}_{FOLEQ}(SP/\approx)}$ is sound.
  Assume that $SP \in SPEX(RBASL)$ and assume that $\vdash_{\Pi^{RBASL}_{FOLEQ}(SP)}$ is sound. To prove that $\vdash_{\Pi^{RBASL}_{FOLEQ}(SP/\approx)}$ is sound, we have to define for any $A \in Models(SP/\approx)$, a $Symbols(\Pi^{RBASL}_{FOLEQ}(SP/\approx))$-algebra B such that:
  - $B|_{Signature(SP)} = \mathrm{A}$
  - $\forall \psi \in \Gamma env(\Pi^{ASL}_{AINS}, SP/\approx).B \models \psi$.
  - B satisfies $\Pi^{RBASL}_{FOLEQ}(SP/\approx)$

By the definition of $Models(SP/\approx)$, we know that there exists a $Signature(SP)$-algebra A' such that $A'/\approx^{In,Obs}_{A'} \cong A$ and $A' \in Models(SP)$. In the following, we will denote by $h : A'/\approx^{Obs,In}_{A'} \rightarrow A$ the isomorphism between $A'/\approx^{Obs,In}_{A'}$ and A. By induction hypotheses, we know that there exists a $Symbols(\Pi^{RBASL}_{FOLEQ}(SP))$-algebra A'' such that $A''|_{Signature(SP)} = A'$ and A'' satisfies $\Pi^{RBASL}_{FOLEQ}(SP)$. The $Symbols(\Pi^{RBASL}_{FOLEQ}(SP/\approx))$-algebra B is defined as follows:

$B|_{\Sigma} = A$(and therefore it satisfies the first condition).

$B|_{Copy''(Symbols(\Pi^{RBASL}_{FOLEQ}(SP)))} = A''|_{Copy''-1}$.

$B_{\pi_{Copy(s)}} = \epsilon_{A'|_{Copy-1},Copy(s)}$for every sort $s \in S$.

where

$\epsilon_{A,Copy(s)}(v) = h([v]_{\approx^{Copy(Obs),Copy(In)}_{A,Copy(s)}})$for any A in $Alg(Copy(\Sigma))$,

for any $s \in S$ and for any $v \in A_{Copy(s)}$

$B_{\sim_{Copy(s)}} = \approx^{Copy(In),Copy(Obs)}_{A'|_{Copy-1},Copy(s)}$

$B|_{c[Copy(r),Copy(s)]} =$

$\{\lambda frv : A'|_{Copy-1}[Copy(In)]_{Copy(r)}.I_{\alpha \cup \{(z_{Copy(r)},frv)\}}(czr) \mid$

$\alpha \in (X_{Copy(In)} \rightarrow A')\,,\ czr \in T_{Copy(\Sigma),Copy(s)}(X_{Copy(In)} \cup \{z_{Copy(r)}\})\}$

for every sort $s, r \in S$ and for a S-sorted infinite enumerable set of variables such that for every sort $s \in S$ $X_s \cap z_s = \emptyset$.

$c[z_{Copy(s)}]_B = z_{Copy(s)}$for every sort $s \in S$

$c[Copy(r), Copy(f)]_B(cx_1, \ldots, cx_n) = Copy(f)(cx_1, \ldots, cx_n)$

for every sort $r \in S$, $f : s_1 \times \ldots \times s_n \to s \in Op$,

$cx_1 \in T_{Copy(\Sigma),Copy(s_1)}(X_{Copy(In)} \cup \{z_{Copy(r)}\}), \ldots ,$

$cx_n \in T_{Copy(\Sigma),Copy(s_n)}(X_{Copy(In)} \cup \{z_{Copy(r)}\})$

$c\_appl[Copy(r), Copy(s)]_B$

$(\lambda frv : A'[Copy(In)]_{Copy(r)}.I_{\alpha \cup \{(z_{Copy(r)}, frv)\}}(czr), rv) \;=$

$I_{\alpha \cup \{(z_{Copy(r)}, rv)\}}(czr)$ for all $\alpha : X_{Copy(In)} \to A[Copy(In)]$,

for all $czr \in T_{Copy(\Sigma),Copy(s)}(X_{Copy(In)} \cup \{z_{Copy(r)}\})$

and for all $rv \in A[Copy(In)]_{Copy(s)}$.

To show that

$$\forall \psi \in \Gamma env(\Pi^{ASL}_{AINS}, SP/\approx).B \models \psi$$

we have to show the following propositions:

- $\forall \psi \in \Gamma env(\Pi^{ASL}_{AINS}, rename \quad SP \quad \textbf{by} \quad Copy).B \models \psi$ which follows by the induction hypotheses and the proof of soundness for the rename operator.
- $B \models pEpi_{FOLEQ}[Copy(\Sigma)[\sim, \pi_{Copy^{-1}}, z, c, c\_appl]] \wedge$

  $B \models Ax\_c\_appl[Copy(\Sigma)[\sim, \pi_{Copy^{-1}}, z, c, c\_appl]]$

  $B \models pEpi_{FOLEQ}[Copy(\Sigma)[\sim, \pi_{Copy^{-1}}, z, c, c\_appl]]$ because for every sort s $B_{\pi_{Copy(s)}}$ can be seen as an epimorphism between A' and A and therefore B satisfies $Hom[Copy(\Sigma)[\sim, \pi_{Copy}]$, $Epihom[Copy(\Sigma)[\sim, \pi_{Copy}]]$ and also $\sim -comp[Copy(\Sigma)[\sim, \pi_{Copy}]]$. The proof of $B \models Ax\_c\_appl[\Sigma[\sim, \pi_{Copy}, z, c, c\_appl]]$ is straightforward.

And to show that B satisfies $\Pi^{RBASL}_{FOLEQ}(SP/\approx)$ we have to show the following propositions:

- $B|_{Copy''(Symbols(\Pi^{RBASL}_{FOLEQ}(SP)))}$ satisfies $\Pi^{RBASL}_{FOLEQ}(rename \quad SP \quad \textbf{by} \quad Copy)$. Since $A' \in Models(SP)$, by induction hypotheses we have that A'' satisfies $\Pi^{RBASL}_{FOLEQ}(SP)$ where $A''|_{Signature(SP)} = A'$. Therefore, $B|_{Copy''(Symbols(\Pi^{RBASL}_{FOLEQ}(SP)))}$ satisfies $\Pi^{RBASL}_{FOLEQ}(rename \quad SP \quad \textbf{by} \quad Copy)$.
- B satisfies $Indist\_rel\_sr[Copy(\Sigma)[\sim, \pi_{Copy^{-1}}, z, c, c\_appl], Copy(Obs), Copy(In)]$ in the same way as in the case of the behavioural operator.

- For all specification expressions $SP \in SPEX(RBASL)$ such that $Signature(SP) = \Sigma = (S, Op)$, if $\vdash_{\Pi^{RBASL}_{FOLEQ}(SP)}$ is sound then $\vdash_{\Pi^{RBASL}_{FOLEQ}(\textbf{abstract} \quad SP \quad \textbf{by} \quad \equiv)}$ is sound.
  It is trivial since by the two previous proofs we know that if $\vdash_{\Pi^{RBASL}_{FOLEQ}(SP)}$ is sound then $\vdash_{\Pi^{RBASL}_{FOLEQ}(\textbf{behaviour} \quad SP/\approx \quad \textbf{wrt} \quad \approx)}$ is sound.

Theorem 2.25 *There is no sound and complete consequence relation of the form* $\vdash_{\Pi^{RBASL}_{FOLEQ}}$ *for RBASL.*

Proof:
See [MS85]. The basic idea of the proof is to define a specification SP such that the set $\{\phi \mid SP \models \phi\}$ is not recursively enumerable since they work with consequence relations $\vdash$ which are recursively enumerable.They show that the set of sentences which satisfy the single sorted algebra NAT with carrier set the natural numbers and with the usual operations $zero, succ, substract, add, multiply$ (where $substract_{NAT}(m, n) = 0$ if $n > m$) is not recursively enumerable. They use equational logic with data constraints to define an specification whose class of models is the class of algebras isomorphic to NAT. Since all the consequence relations of our proof systems are recursively enumerable because we work with natural deduction systems with a finite set of finitary rules, the proof is valid for our framework if we define the specification in RBASL. We will refer to this specification as NSP and it is defined as follows:

$$NSP = reach\ Nat\ with\ (nat, (zero : nat, succ : nat \to nat))$$

where

$$Nat = < (nat, zero : nat, succ : nat \to nat, add : nat \times nat \to nat,$$

$$substract : nat \times nat \to nat, multiply : nat \times nat \to nat),$$

$$(\forall m, n : nat.add(n, zero) = n \wedge add(n, succ(m)) = succ(add(n, m)) \wedge$$

$$(\forall m, n : nat.substract(zero, n) = zero \wedge$$

$$substract(succ(n), succ(m)) = substract(n, m)) \wedge$$

$$(\forall m, n : nat.multiply(n, zero) = zero \wedge$$

$$multiply(n, succ(m)) = add(n, multiply(n, m))$$

$$(\forall m, n : nat.\neg(n = n + succ(m))) >$$

It is trivial to show that

$$Models(NSP) = \{A \mid A \cong NAT\}$$

since one can prove that there exists an isomorphism between any model of NSP and the initial model of NSP which is isomorphic to NAT.

2.1.3 The proof system Π_{HOL}^{RBASL}

Before presenting the proof system Π_{HOL}^{RBASL} associated to the specification language $RBASL$, we present the proof system of the higher order logic HOL $\Pi_{HOL}(\Gamma)$. This proof system is split in three different kind of rules: derivation rules, typing rules and proof rules which determine the β-equality. All of them are standard rules presented in a natural deduction style.

Definition 2.26 *The natural deduction system $\Pi_{HOL}(\Gamma)$ is defined by the following set of rules for any $\Gamma \in \mathcal{P}(|Sen_{AINS}(\Sigma)|)$:*

$$\frac{\Gamma \cup \phi \Rightarrow_X \phi'}{\Gamma \Rightarrow_X \phi \supset \phi'} \quad (\supset\ i)$$

$$\frac{\Gamma \Rightarrow_X \phi \supset \phi' \quad \Gamma \Rightarrow_X \phi}{\Gamma \Rightarrow_X \phi'} \quad (\supset\ e)$$

$$\frac{\Gamma \Rightarrow_{X \cup x:\tau} \phi}{\Gamma \Rightarrow_X \forall x : \tau.\phi} \quad (\forall\ i)$$

$$\frac{\Gamma \Rightarrow_X \forall x : \tau.\phi \quad X \blacktriangleright t : \tau}{\Gamma \Rightarrow_X \phi\{t/x\}} \quad (\forall\ e)$$

$$\frac{\Gamma \Rightarrow_X \phi \quad \phi\ =_{X,\beta}\ \phi'}{\Gamma \Rightarrow_X \psi} \quad (CONV)$$

where the set of typing rules of this proof system is:

$$\frac{}{X \blacktriangleright x_\tau : \tau}\ x_\tau \in X_\tau \quad (ASS)$$

$$\frac{X \blacktriangleright t_1 : s_1 \quad \ldots \quad X \blacktriangleright t_n : s_n}{X \blacktriangleright f(t_1, \ldots, t_n) : s}\ f : s_1 \times \ldots \times s_n \to s \in \Sigma \quad (APPL)$$

$$\frac{X \cup \{x_1 : \tau_1, \ldots, x_n : \tau_n\} \blacktriangleright \phi : \mathbf{Prop}}{X \blacktriangleright \lambda(x_1 : \tau_1, \ldots, x_n : \tau_n).\phi : [\tau_1, \ldots, \tau_n]} \quad (\lambda ABS)$$

$$\frac{X \blacktriangleright t_1 : \tau_1 \quad \ldots \quad X \blacktriangleright t_n : \tau_n \quad X \blacktriangleright t : [\tau_1, \ldots, \tau_n]}{X \blacktriangleright t(t_1, \ldots, t_n) : \mathbf{Prop}} \quad (\lambda APPL)$$

$$\frac{X \cup x : \tau \blacktriangleright \phi : \mathbf{Prop}}{X \blacktriangleright \forall x : \tau.\phi : \mathbf{Prop}} \quad (\forall)$$

$$\frac{X \blacktriangleright \phi : \mathbf{Prop} \quad X \blacktriangleright \phi' : \mathbf{Prop}}{X \blacktriangleright \phi \supset \phi' : \mathbf{Prop}} \quad (\supset)$$

and the β-equality rules are:

$$\frac{}{x_\tau =_{\beta,X} x_\tau}\ x \in X_\tau \qquad (Vareq)$$

$$\frac{\begin{array}{c} X \blacktriangleright t_1 : s_1 \quad \ldots \quad X \blacktriangleright t_n : s_n \\ t_1 =_{\beta,X} t'_1 \ldots t_n =_{\beta,X} t'_n \end{array}}{f(t_1,\ldots,t_n) =_{\beta,X} f(t'_1,\ldots,t'_n)}\ f : s_1 \times \ldots \times s_n \to s_n \in \Sigma \qquad (Termeq)$$

$$\frac{\begin{array}{c} X \blacktriangleright t\,(t_1,\ldots,t_n) : \mathbf{Prop} \quad t =_{\beta,X} t' \\ t_1 =_{\beta,X} t'_1 \ldots t_n =_{\beta,X} t'_n \end{array}}{t(t_1,\ldots,t_n) =_{\beta,X} t'(t'_1,\ldots,t'_n)}\ (Appleq)$$

$$\frac{\begin{array}{c} X \blacktriangleright t_1 : \tau_1 \quad \ldots \quad X \blacktriangleright t_n : \tau_n \\ X \blacktriangleright \lambda(x_1 : \tau_1,\ldots,x_n : \tau_n).\phi : [\tau_1,\ldots,\tau_n] \\ \phi\ \{t_1/x_1\} \ldots \{t_n/x_n\} =_{\beta,X} \phi' \end{array}}{\lambda(x_1 : \tau_1,\ldots,x_n : \tau_n).\phi\ (t_1,\ldots,t_n) =_{\beta,X} \phi'}\ (Llambdaeq)$$

$$\frac{\begin{array}{c} X \blacktriangleright \lambda(x_1 : \tau_1,\ldots,x_n : \tau_n).\phi : [\tau_1,\ldots,\tau_n] \\ \phi =_{\beta,X\cup\{x_1:\tau_1,\ldots,x_n:\tau_n\}} \phi'\ \{x_1/x'_1\} \ldots \{x_n/x'_n\} \end{array}}{\lambda(x_1 : \tau_1,\ldots,x_n : \tau_n).\phi =_{\beta,X} \lambda(x'_1 : \tau_1,\ldots,x'_n : \tau_n).\phi'}\ (Lambdaeq)$$

$$\frac{\begin{array}{c} X \cup \{x : \tau\} \blacktriangleright \phi : \mathbf{Prop} \\ \phi =_{\beta,X\cup\{x:\tau\}} \phi'\{x/x'\} \end{array}}{\forall x : \tau.\phi =_{\beta,X} \forall x' : \tau.\phi'}\ (Foralleq)$$

$$\frac{\begin{array}{c} X \blacktriangleright \phi' : \mathbf{Prop} \\ X \blacktriangleright \phi : \mathbf{Prop} \\ \phi' =_{\beta,X} \phi \end{array}}{\phi =_{\beta,X} \phi'}\ (Sym)$$

Definition 2.27 *The encoding of the logical operators* **false**, **true**, $=, \vee, \wedge, \exists$ *is as follows:*

$$\begin{array}{l} false =_{def} \forall P : \mathbf{Prop}.P \\ true =_{def} \forall P : \mathbf{Prop}.P \supset P \\ t =_\tau r =_{def} \forall P : [\tau].P\ t \supset P\ r \\ \phi \wedge \phi' =_{def} \forall P : \mathbf{Prop}.(\phi \supset \phi' \supset P) \supset P \\ \phi \vee \phi' =_{def} \forall P : \mathbf{Prop}.(\phi \supset P) \supset (\phi' \supset P) \supset P \\ \exists x : \tau.\phi =_{def} \forall P : \mathbf{Prop}.((\forall x : \tau.\phi) \supset P) \supset P \end{array}$$

Proposition 2.28 *The following set of rules is admissible in the proof system* $\Pi_{HOL}(\Gamma)$ *for any environment* $\Gamma \in \mathcal{P}(|Sen_{AINS}(\Sigma)|)$*:*

$$\frac{}{\{\} \Rightarrow_X \mathbf{true}} \quad (T) \qquad \frac{}{\Gamma \Rightarrow_X \mathbf{false} \supset \phi} \quad (F)$$

$$\frac{\Gamma \Rightarrow_X \phi_1 \wedge \phi_2}{\Gamma \Rightarrow_X \phi_1} \quad (\wedge El) \qquad \frac{\Gamma \Rightarrow_X \phi_1 \wedge \phi_2}{\Gamma \Rightarrow_X \phi_2} \quad (\wedge Er)$$

$$\frac{\Gamma \Rightarrow_X \phi_1 \quad \Gamma \Rightarrow_X \phi_2}{\Gamma \Rightarrow_X \phi_1 \wedge \phi_2} \quad (\wedge I)$$

$$\frac{\Gamma \Rightarrow_X \phi_1}{\Gamma \Rightarrow_X \phi_1 \vee \phi_2} \quad (\vee Il) \qquad \frac{\Gamma \Rightarrow_X \phi_2}{\Gamma \Rightarrow_X \phi_1 \vee \phi_2} \quad (\vee Ir)$$

$$\frac{\Gamma \Rightarrow_X \phi_1 \vee \phi_2 \quad \Gamma \Rightarrow_X \phi_1 \supset \psi \quad \Gamma \Rightarrow_X \phi_2 \supset \psi}{\Gamma \Rightarrow_X \psi} \quad (\vee E)$$

$$\frac{X \blacktriangleright t : \tau \quad \Gamma \cup \{t : \tau\} \Rightarrow_X \phi\{t/x\}}{\Gamma \Rightarrow_X \exists x : \tau.\phi} \quad (\exists I)$$

$$\frac{\Gamma \Rightarrow_X \exists x : \tau.\phi \quad \Gamma \cup \{\phi\} \Rightarrow_{X \cup \{x:\tau\}} \psi}{\Gamma \Rightarrow_X \psi} \quad (\exists E)$$

$$\frac{\Gamma \cup \{\phi\} \Rightarrow_X \mathbf{false}}{\Gamma \Rightarrow_X \neg\phi} \quad (\neg I) \qquad \frac{\Gamma \Rightarrow_X \psi \quad \Gamma \Rightarrow_X \neg\psi}{\Gamma \Rightarrow_X \phi} \quad (\neg E)$$

In the next definitions, we present different relational signatures which extend a given signature with the following symbols:

- Symbols to denote the indistinguishability relation for every sort of the signature.
- The same extension as in the previous one plus symbols to denote a pseudoepimorphism between the original signature and a disjoint copy.

Definition 2.29 *The relational signature* $\Sigma[\sim]$ *is defined for any signature* $\Sigma = (S, Op)$ *and for any S-family of new symbols* $\sim$ *as:*

$$\Sigma[\sim] = \Sigma \cup \{\sim_s : s \times s | s \in S\}$$

Definition 2.30 *The relational signature* $\Sigma[\sim, \pi_{Copy}]$ *for any signature* $\Sigma = (S, Op)$*, for any bijective signature morphism* $Copy : \Sigma \to Copy(\Sigma)$ *such that* $\Sigma \cap Copy(\Sigma) = \emptyset$ *and for any* $S-$*family of new symbols* $\sim$ *and* π *is defined as:*

$$\Sigma[\sim, \pi_{Copy}] = \Sigma[\sim] \cup Copy(\Sigma) \cup \{\pi_s : s \to Copy(s) | s \in S\}$$

Remark: *The relational signature* $Copy(\Sigma)[\sim, \pi_{Copy^{-1}}]$ *stands for the following signature:*

$$Copy(\Sigma)[\sim, \pi_{Copy^{-1}}] = Copy(\Sigma)[\sim] \cup \Sigma \cup \{\pi_s : Copy(s) \to s | s \in S\}$$

Here we present some axioms used to define the proof system for the reachability operator:

Definition 2.31 *The set of sentences* $Reach\_ax[\Sigma, (\mathcal{S}_\mathcal{R}, \mathcal{F}_\mathcal{R})]$ *for any reachability constraint* $(\mathcal{S}_\mathcal{R}, \mathcal{F}_\mathcal{R})$ *of* Σ *is defined as follows:*

$$Reach\_ax[\Sigma, (\mathcal{S}_\mathcal{R}, \mathcal{F}_\mathcal{R})] = \{\forall P : [s].(\bigwedge_{f:s_1 \times \ldots \times s_n \to s \in \mathcal{F}_\mathcal{R}} (\forall x_1 : s_1 \ldots . \forall x_n : s_n.$$

$$(\bigwedge_{i \in [1..n], s_i = s} P(x_i) \supset P(f(x_1, \ldots, x_n))) \supset \forall x : s.P(x) \mid s \in \mathcal{F}_\mathcal{R} \}$$

In the following, there are some definitions which axiomatise the indistinguishability relation in higher-order logic. This axiomatisation contains the following axioms:

- Axioms to determine which a given value is reachable.
- Axioms to determine which a given relation coincide with the set-theoretical equality.
- Axioms to determine that a family of relations is a congruence.

Definition 2.32 *The* $Sorts(\Sigma)$*-family of sentences* $D[\Sigma, In]$ *for any* $In \subseteq Sorts(\Sigma)$ *is defined as follows:*

$$D[\Sigma, In] = \{D_s[\Sigma, In] \mid s \in Sorts(\Sigma)\}$$

where

$$D_s[\Sigma, In] = \lambda x : s.(\forall P : [s].(\bigwedge_{f:s_1 \times \ldots \times s_n \to s \in Ops(\Sigma)} (\forall x_1 : s_1 \ldots . \forall x_n : s_n.$$

$$(\bigwedge_{i \in [1..n], s_i = s} P(x_i) \supset P(f(x_1, \ldots, x_n))))) \supset P\, x) \qquad , if\ s \in S - In$$

$$D_s[\Sigma, In] = \lambda x : s.\mathbf{true} \qquad , if\ s \in In$$

Definition 2.33 *The set of sentences* $Indist\_rel[(S,Op)[\sim],Obs,In]$ *is defined as follows:*

$$Indist\_rel[(S,Op)[\sim],Obs,In] = \{\ Indist\_rel[(S,Op)[\sim],Obs,In,s]\ \mid\ s \in S\}$$

where

$$Indist\_rel[(S,Op)[\sim],Obs,In,s] = \lambda x : s.\lambda y : s.\exists_{s' \in S} R_{s'} : [s',s'].$$

$$R_s(x,y)\ \wedge OBSEQ[(S,Op),R,Obs,In]\ \wedge$$

$$CONG[(S,Op),R,Obs,In]\ \wedge\ D_s[\Sigma,In](x)\ \wedge\ D_s[\Sigma,In](y)$$

$$OBSEQ[(S,Op),R,Obs,In] = \bigwedge_{obs \in Obs} \forall v : obs.\forall w : obs.$$

$$D_{obs}[\Sigma,In](v)\ \wedge D_{obs}[\Sigma,In](w)\ \Rightarrow (R_{obs}(v,w) \Leftrightarrow v = w)$$

$$CONG[(S,Op),R,Obs,In] =$$

$$\bigwedge_{f:s_1 \times \ldots \times s_n \to s \in Op} \forall x_1 : s_1.\forall y_1 : s_1.\ \ldots\ .\forall x_n : s_n.\forall y_n : s_n.$$

$$(R_{s1}(x_1,y_1)\ \wedge\ \ldots\ \wedge\ R_{sn}(x_n,y_n)\ \Rightarrow$$

$$R_s(f(x_1,\ldots,x_n),f(y_1,\ldots,y_n)))$$

Here, we present an axiomatisation of a pseudoepimorphism over the signature $\Sigma[\sim,\pi_{Copy}]$ in higher-order logic. This axiomatisation is equivalent to the one presented in first-order logic.

Definition 2.34 *The set of sentences* $pEpi_{HOL}[\Sigma[\sim,\pi_{Copy}],Obs,In]$ *is defined as:*

$$pEpi_{HOL}[\Sigma[\sim,\pi_{Copy}],Obs,In]\ =\ Hom[\Sigma[\sim,\pi_{Copy}],Obs,In] \cup$$

$$Epihom[\Sigma[\sim,\pi_{Copy}],Obs,In]\ \cup\ \sim\!-comp[\Sigma[\sim,\pi_{Copy}],Obs,In]$$

where

$$Hom[\Sigma[\sim,\pi_{Copy}],Obs,In] = \bigcup_{f:s_1 \times \ldots \times s_n \to s \in Op} \{\forall t_1 : s_1.\ \ldots\ .\forall t_n : s_n.$$

$$\bigwedge_{i \in [1..n]} D_{s_i}[\Sigma,In](t_i)\ \Rightarrow$$

$$\pi_{Copy,s}(f(t_1,\ldots,t_n)) = Copy(f)(\pi_{Copy,s_1}(t_1),\ldots,\pi_{Copy,s_n}(t_n))\}$$

$$Epihom[\Sigma[\sim, \pi_{Copy}], Obs, In] =$$

$$\bigcup_{s \in S} \{\forall y : Copy(s).\exists x : s.D_s[\Sigma, In](x) \ \wedge \ \pi_{Copy,s}(x) = y)\}$$

$$\sim -comp[\Sigma[\sim, \pi_{Copy}], Obs, In] =$$

$$\bigcup_{s \in S} \{\forall x : s.\forall y : s.D_s[\Sigma, In](x) \wedge D_s[\Sigma, In](y) \Rightarrow$$

$$x \sim_s y \Leftrightarrow \pi_{Copy,s}(x) \ = \ \pi_{Copy,s}(y)\}$$

In the following definitions, we define specification expressions using an extended version of $RBASL$ with relational signatures and we prove some equivalences between the semantics of some of these specification expressions and the semantics of the behavioural operator. The semantics of $RBASL$ with relational signatures is extended in the obvious way basically just replacing signatures by relational signatures. The specification expressions defined below are used to establish an equivalence between the semantics of behavioural operators and structured specifications which are the basis to define the proof system Π_{HOL}^{RBASL}.

Definition 2.35 *The specification expression* $BSP[\Sigma[\sim, \pi_{Copy}], Obs, In]$ *for any signature* $\Sigma = (S, Op)$*, for any bijective signature morphism* $Copy : \Sigma \rightarrow Copy(\Sigma)$ *such that* $\Sigma \cap Copy(\Sigma) = \emptyset$ *and for any* $S-$*family of new symbols* $\sim$ *and* π *is defined as:*

$$BSP[\Sigma[\sim, \pi_{Copy}], Obs, In] =< \Sigma[\sim, \pi_{Copy}],$$

$$\{\forall x : s.\forall y : s.x \sim_s y \Leftrightarrow Indist\_rel[\Sigma[\sim], Obs, In, s](x, y) \mid s \in S\} \ \cup$$

$$pEpi[\Sigma[\sim, \pi_{Copy}], Obs, In] >$$

Definition 2.36 *For any specification expression* SP *with signature* $\Sigma = (S, Op)$*, for any signature morphism* $Copy : \Sigma \rightarrow Copy(\Sigma)$ *and* $Copy' : \Sigma \rightarrow Copy'(\Sigma)$ *such that* $\Sigma \cap Copy(\Sigma) = \emptyset$ *and* $\Sigma \cap Copy'(\Sigma) = \emptyset$*, for any* $S-$*family of new symbols* $\sim$ *and* π*, the specification expression* $GBSP[SP, Copy, \Sigma[\sim, \pi_{Copy'}]]$ *is defined as:*

$$GBSP[SP, Copy, \Sigma[\sim, \pi_{Copy'}], Obs, In] =$$

$$rename \quad SP \quad \textbf{by} \quad Copy \ + \ BSP[\Sigma[\sim, \pi_{Copy'}], Obs, In]$$

Lemma 2.37 *Let $R_A^{Obs,In}$ be a S-family of partial congruences which satisfies the following condition:*

$$\forall obs \in Obs.\forall v, w \in A_{obs}[X_{In}].(v\ R_{A,obs}^{Obs,In}\ w \ \Leftrightarrow\ v = w)$$

then

$$\forall s \in S.\forall v, w \in A[X_{In}].v\ R_{A,s}^{Obs,In}\ w \ \Rightarrow\ v \approx_{A,s}^{Obs,In} w$$

Proof sketch:
It follows by context induction. The proof of the general case uses that R is a S-family of partial congruences which coincides with the set theoretical equality for observable sorts.

The following lemma can also be found in [HS96]:

Lemma 2.38 *The sentence $Indist\_rel[(S,Op)[\sim], Obs, In, s]$ for any sort $s \in S$ and for any free variables $x, y \in X_s$ satisfies the following condition which we will refer as the indistinguishability condition:*

$$\forall s \in S.\forall A \in Alg(\Sigma).\forall \rho \in \{x, y\} \ \rightarrow\ A.$$

$$[\![Indist\_rel[\Sigma[\sim], Obs, In](x, y)]\!]_{\rho,A} \Leftrightarrow \rho(x) \approx_{A,s}^{Obs,In} \rho(y)$$

Theorem 2.39 *For any specification expression SP with signature $\Sigma = (S, Op)$, for any signature morphism $Copy : \Sigma \rightarrow Copy(\Sigma)$ such that $\Sigma \cap Copy(\Sigma) = \emptyset$, for any $S-$family of symbols $\sim$ and π the two following equivalences between specification expressions hold:*

$$\textbf{behaviour}\ \ SP\ \ \textbf{wrt}\ \ \approx\ \ = GBSP[SP, Copy, \Sigma[\sim, \pi_{Copy}], Obs, In]|_\Sigma$$

$$SP/\approx\ \ = GBSP[SP, Copy, Copy(\Sigma)[\sim, \pi_{Copy^{-1}}], Copy(Obs), Copy(In)]|_\Sigma$$

Proof:

- **behaviour** SP **wrt** $\approx$ $= GBSP[SP, Copy, \Sigma[\sim, \pi_{Copy}], Obs, In]|_\Sigma$.
  Assume that $BEHSP = BSP[\Sigma[\sim, \pi_{Copy}], Obs, In]$. We differentiate two cases:
  - $A \in Models($**behaviour** SP **wrt** $\approx) \Rightarrow$
    $A \in Models(GBSP[SP, Copy, \Sigma[\sim, \pi_{Copy}], Obs, In]|_\Sigma)$.
    Assume that $A \in Models($**behaviour** SP **wrt** $\approx)$. The proposition which we have to prove is equivalent to

    $$\exists B \in Models(GBSP[SP, Copy, \Sigma[\sim, \pi_{Copy}], Obs, In]).B|_\Sigma = A$$

    To prove this proposition we will build an algebra $B \in Models(GBSP[SP, Copy, \Sigma[\sim, \pi_{Copy}], Obs, In])$ such that $B|_\Sigma = A$. The algebra B is defined as follows:

* $B|_{\Sigma} = A$.
* $B|_{Copy(\Sigma)} = A/\approx_A^{Obs,In}|_{Copy-1}$.
* $B_{\pi_s} = \epsilon_{A,s}$ for every sort $s \in S$.
  where
  $\epsilon_{A,s}(v) = [v]_{\approx_{A,s}^{Obs,In}}$ for all $v \in A_s$.
* $B_{\sim_s} = \approx_{A,s}^{In,Obs}$ for every sort $s \in S$

To show that $B \in Models(GBSP[SP, Copy, \Sigma[\sim, \pi_{Copy}], Obs, In])$ we have to show that $B|_{Copy(\Sigma)}|_{Copy} \in Models(SP)$ which obviously holds since $B|_{Copy(\Sigma)}|_{Copy} = A/\approx_A^{Obs,In}$ and $A/\approx_A^{Obs,In} \in Models(SP)$.
We have to show also that

$$B \in Models(< \Sigma[\sim, \pi_{Copy}], \\ Indist\_rel[\Sigma[\sim], Obs, In] \cup pEpi[\Sigma[\sim, \pi_{Copy}], Obs, In] >)$$

This proposition holds because $B \models Indist\_rel[\Sigma[\sim], Obs, In]$ and $B \models pEpi[\Sigma[\sim, \pi_{Copy}], Obs, In]$.

$B \models Indist\_rel[\Sigma[\sim], Obs, In]$ holds since for every sort s $B_{\sim_s} = \approx_{B|_{\Sigma},s}^{In,Obs}$ and $Indist\_rel[\Sigma[\sim], Obs, In]$ satisfies the indistinguishability condition, and $B \models pEpi[\Sigma[\sim, \pi_{Copy}]]$ holds since for every sort s, B_{π_s} can be seen as an epimorphism between A and $A/\approx_{A,s}^{Obs,In}$ and therefore B satisfies $Hom[\Sigma[\pi_{Copy}, Obs, In], Epihom[\Sigma[\pi_{Copy}], Obs, In]$ and also $\sim -comp[\Sigma[\sim, \pi_{Copy}], Obs, In]$.

– $A \in Models(GBSP[SP, Copy, \Sigma[\sim, \pi_{Copy}], Obs, In]|_{\Sigma}) \Rightarrow$
$A \in Models(\textbf{behaviour} \quad SP \quad \textbf{wrt} \quad \approx)$.

Assume that $A \in Models(GBSP[SP, Copy, \Sigma[\sim, \pi_{Copy}], Obs, In]|_{\Sigma})$.
This proposition can be rewritten to

$$\exists B \in Models(GBSP[SP, Copy, \Sigma[\sim, \pi_{Copy}], Obs, In]).B|_{\Sigma} = A$$

Let B be a $Signature(GBSP[SP, Copy, \Sigma[\sim, \pi_{Copy}], Obs, In])$-algebra such that $B \in Models(GBSP[SP, Copy, \Sigma[\sim, \pi_{Copy}], Obs, In])$ and $B|_{\Sigma} = A$. By the definition of $Models(GBSP[SP, Copy, \Sigma[\sim, \pi_{Copy}], Obs, In])$ we have that $B \in Models(BEHSP)$ and because of the indistinguishability condition together with the definition of $pEpi[\Sigma[\sim, \pi_{Copy}]]$ we have that

$$\exists C \in Alg(Copy(\Sigma)).C|_{Copy} \cong A/\approx_A^{Obs,In} \;\&\; B|_{Copy(\Sigma)} = C$$

Let C be a $Copy(\Sigma)$-algebra such that $C|_{Copy} \cong A/\approx_A^{Obs,In}$ and $B|_{Copy(\Sigma)} = C$. By the definition of $Models(GBSP[SP, Copy, \Sigma[\sim, \pi_{Copy}], Obs, In]|_{\Sigma})$ and since $B|_{Copy(\Sigma)} = C$ we have that $C \in$

$Models(rename \quad SP \quad \textbf{by} \quad Copy)$ and since $C|_{Copy} \cong A/\approx_A^{Obs,In}$ and SP is closed under isomorphism we have that $A/\approx_A^{Obs,In} \in Models(SP)$ and therefore

$$A \in Models(\textbf{behaviour} \quad SP \quad \textbf{wrt} \quad \approx)$$

- $SP/\approx = GBSP[SP, Copy, Copy(\Sigma)[\sim, \pi_{Copy-1}], Copy(Obs), Copy(In)]|_\Sigma$.
  Assume that
  $QGBSP = GBSP[SP, Copy, Copy(\Sigma)[\sim, \pi_{Copy-1}], Copy(Obs), Copy(In)]|_\Sigma$.
  We differentiate two cases:
  - $A \in Models(SP/\approx) \Rightarrow A \in Models(QGBSP)$

    Assume that $A \in Models(SP/\approx)$. The proposition which we have to prove is equivalent to

    $$\exists B \in Models(QGBSP) \,.\, B|_\Sigma = A$$

    To prove this proposition we will build an algebra

    $$B \in Models(QGBSP)$$

    such that $B|_\Sigma = A$. By the proposition $A \in Models(SP/\approx)$ we know that

    $$\exists A' \in Alg(Signature(SP)).A'/\approx_{A'}^{Obs,In} \cong A \;\&\; A' \in Models(SP)$$

    Let A' be a $Signature(SP)$-algebra such that $A'/\approx_A^{Obs,In} \cong A$ and $A' \in Models(SP)$ and let $h : A'/\approx_{A'}^{Obs,In} \to A$ be the isomorphism which relates each other. The algebra B is defined as follows:
    - $B|_\Sigma = A$.
    - $B|_{Copy(\Sigma)} = A'|_{Copy-1}$.
    - $B_{\pi_{Copy(s)}} = \epsilon_{A'|_{Copy-1}, Copy(s)}$ for every sort $s \in S$.
      where
      $\epsilon_{A,Copy(s)}(v) = h([v]_{\approx_{A,Copy(s)}^{Copy(Obs),Copy(In)}})$
    - $B_{\sim_{Copy(s)}} = \approx_{A'|_{Copy-1},Copy(s)}^{Copy(In),Copy(Obs)}$

    To show that $B \in Models(QGBSP)$ we have to show that $B|_{Copy(\Sigma)}|_{Copy} \in Models(SP)$ which obviously holds since $B|_{Copy(\Sigma)}|_{Copy} = A'$ and we have to show also that

    $$B \in Models(BSP[Copy(\Sigma)[\sim, \pi_{Copy-1}], Copy(Obs), Copy(In)])$$

    which holds since $B \models Indist\_rel[Copy(\Sigma)[\sim], Obs, In]$ and $B \models pEpi[\Sigma[\sim, \pi_{Copy-1}]]$.

$B \models Indist\_rel[Copy(\Sigma)[\sim], Obs, In]$ holds since for every sort s $B_{\sim_{Copy(s)}} = \approx^{Copy(In),Copy(Obs)}_{B|_{Copy(\Sigma)},Copy(s)}$ and $Indist\_rel[Copy(\Sigma)[\sim], Copy(Obs), Copy(In)]$ satisfies the indistinguishability condition. $B \models pEpi[Copy(\Sigma)[\sim, \pi_{Copy-1}]]$ holds since for every sort s, $B_{\pi_{Copy(s)}}$ can be seen as an epimorphism between A' and A and therefore B satisfies $Hom[\Sigma[\sim, \pi_{Copy}], Obs, In]$, $Epihom[\Sigma[\pi_{Copy}], Obs, In]$ and also $\sim -comp[\Sigma[\sim, \pi_{Copy}], Obs, In]$.

- $A \in Models(QGBSP) \Rightarrow A \in Models(SP/\approx)$. Assume that $A \in Models(QGBSP)$. By this proposition we know that there exists a $Signature(QGBSP)$-algebra B such that $B \in Models(QGBSP)$ and $B|_\Sigma = A$. By the definition of $Models(QGBSP)$ we have that $B|_{Copy(\Sigma)}|_{Copy} \in Models(SP)$ and together with the definitions of $Indist\_rel[Copy(\Sigma)[\sim], Copy(Obs), Copy(In)]$ and $pEpi[Copy(\Sigma)[\sim, \pi_{Copy-1}], Copy(Obs), Copy(In)]$ we have that $A \cong B|_{Copy(\Sigma)}|_{Copy}/\approx^{Obs,In}_A$. Therefore, by the definition of $Models(SP/\approx^{Obs,In}_A)$ we have that $A \in Models(SP/\approx^{Obs,In}_A)$.

Definition 2.40 *The proof system Π^{RBASL}_{HOL} is inductively defined for the specific operators of the language as follows:*

$$Rules(\Pi^{RBASL}_{HOL}, \mathit{rename}\quad SP\quad \textbf{by}\quad \sigma) = \sigma''(Rules(\Pi^{RBASL}_{HOL}, SP))$$

$$Symbols(\Pi^{RBASL}_{HOL}, \mathit{rename}\quad SP\quad \textbf{by}\quad \sigma) =$$

$$\sigma''(Symbols(\Pi^{RBASL}_{HOL}, SP))$$

$$\Gamma env(\Pi^{RBASL}_{HOL}, \mathit{rename}\quad SP\quad \textbf{by}\quad \sigma) = \sigma''(\Gamma env(\Pi^{RBASL}_{HOL}, SP))$$

$$Rules(\Pi^{RBASL}_{HOL}, \textbf{reach}\quad SP\quad \textbf{with}\quad (\mathcal{S}_\mathcal{R}, \mathcal{F}_\mathcal{R})) = Rules(\Pi^{RBASL}_{HOL}, SP)$$

$$Symbols(\Pi^{RBASL}_{HOL}, \textbf{reach}\quad SP\quad \textbf{with}\quad (\mathcal{S}_\mathcal{R}, \mathcal{F}_\mathcal{R})) = Symbols(\Pi^{RBASL}_{HOL}, SP)$$

$$\Gamma env(\Pi^{RBASL}_{HOL}, \textbf{reach}\quad SP\quad \textbf{with}\quad (\mathcal{S}_\mathcal{R}, \mathcal{F}_\mathcal{R})) = \Gamma env(\Pi^{RBASL}_{HOL}, SP) \cup$$

$$Reach\_ax[Signature(SP), (\mathcal{S}_\mathcal{R}, \mathcal{F}_\mathcal{R})]$$

$$Rules(\Pi^{RBASL}_{HOL}, \textbf{behaviour}\quad SP\quad \textbf{wrt}\quad \approx) =$$

$$Copy''(Rules(\Pi^{RBASL}_{HOL}, SP))$$

$$Symbols(\Pi^{RBASL}_{HOL}, \textbf{behaviour}\quad SP\quad \textbf{wrt}\quad \approx) =$$

$$Copy''(Symbols(\Pi^{RBASL}_{HOL}), SP)\ \cup\ \Sigma[\sim, \pi_{Copy}]$$

$$\Gamma env(\Pi^{RBASL}_{HOL}, \textbf{behaviour}\quad SP\quad \textbf{wrt}\quad \approx) =$$

$$\Gamma env(rename\quad SP\quad \textbf{by}\quad Copy) \cup$$

$$\Gamma env(BSP[\Sigma[\sim, \pi_{Copy}], Obs, In])$$

$$Rules(\Pi^{RBASL}_{HOL}, \textbf{abstract}\quad SP\quad \textbf{by}\quad \equiv) =$$

$$Rules(\Pi^{RBASL}_{HOL}, \textbf{behaviour}\quad SP/\approx\quad \textbf{wrt}\quad \approx)$$

$$Symbols(\Pi^{RBASL}_{HOL}, \textbf{abstract}\quad SP\quad \textbf{by}\quad \equiv) =$$

$$Symbols(\Pi^{RBASL}_{HOL}, \textbf{behaviour}\quad SP/\approx\quad \textbf{wrt}\quad \approx)$$

$$\Gamma env(\Pi^{RBASL}_{HOL}, \textbf{abstract}\quad SP\quad \textbf{by}\quad \equiv) =$$

$$\Gamma env(\Pi^{RBASL}_{HOL}, \textbf{behaviour}\quad SP/\approx\quad \textbf{wrt}\quad \approx)$$

$$Rules(\Pi^{RBASL}_{HOL}, SP/\approx) = Copy''(Rules(\Pi^{RBASL}_{HOL}, SP))$$

$$Symbols(\Pi^{RBASL}_{HOL}, SP/\approx) =$$

$$Copy''(Symbols(\Pi^{RBASL}_{HOL}, SP))\ \cup\ Copy(\Sigma)[\sim, \pi_{Copy-1}]$$

$$\Gamma env(\Pi^{RBASL}_{HOL}, SP/\approx) = \Gamma env(\Pi^{RBASL}_{HOL}, rename\quad SP\quad \textbf{by}\quad Copy) \cup$$

$$\Gamma env(BSP[Copy(\Sigma)[\sim, \pi_{Copy-1}], Copy(Obs), Copy(In)])$$

where Σ is the signature of the argument specification SP for every case and the overloaded function symbol σ'' is used here as the pushout of the following

diagram:

$$\begin{array}{ccc} Sym(\Pi^{RBASL}_{HOL}, SP) & \xrightarrow{\sigma''} & PO(i, \sigma'') \\ {\scriptstyle i}\uparrow & & \uparrow \\ \Sigma & \xrightarrow{\quad\sigma\quad} & \Sigma' \end{array}$$

where $i : \Sigma \hookrightarrow Sym(\Pi^{RBASL}_{HOL}, SP)$ *and it is also used as the renaming function of environments and of the proof system* $\Pi^{RBASL}_{HOL}(SP)$ *which use the pushout just described , and the overloaded symbol* $Copy''$ *is the pushout morphism of the following diagram:*

$$\begin{array}{ccc} Sym(\Pi^{RBASL}_{HOL}, SP) & \xrightarrow{Copy''} & PO(i, Copy) \\ {\scriptstyle i}\uparrow & & \uparrow \\ \Sigma & \xrightarrow{\quad Copy\quad} & Copy(\Sigma) \end{array}$$

where $i : \Sigma \hookrightarrow Symbols(\Pi^{RBASL}_{HOL}, SP)$ *and*

$$Copy''(Symbols(\Pi^{RBASL}_{HOL}, SP)) \cap \Sigma[\sim, \pi_{Copy}] \; =$$

$$Copy''(\Sigma)$$

The symbol $Copy''$ *is also used as the renaming functions of environmnents and of set of rules using the pushout morphism just described with the same name.*

Theorem 2.41 *The consequence relation* $\vdash_{\Pi^{RBASL}_{HOL}}$ *is sound.*

Proof:
The proof is by induction of specification expression in a similar way as the proof of soundness of $\vdash_{\Pi^{RBASL}_{FOLEQ}}$ where the proofs for the cases of the behaviour and quotient operator, theorem 2.42 is used.

Theorem 2.42 *There is no sound and complete consequence relation of the form* $\vdash_{\Pi^{RBASL}_{HOL}}$ *for RBASL.*

Proof:
It follows in the same way as the incompleteness of Π^{RBASL}_{FOLEQ}

2.2 The normal form approach

In this subsection, we present the consequence relation $\vdash_{nf,BASLN,\Sigma}$ which relates specification expressions $SP \in SPEX(BASLN)$ of an arbitrary but fixed $BASLnf$ specification language $BASLN$ with a fixed but arbitrary behavioural algebraic institution $BAINS$ with signature $\Sigma \in |AlgSig|$ and sentences $\phi \in |Sen_{BAINS}(\Sigma)|$. The consequence relation is defined using a consequence relation for the behavioural algebraic institution $BAINS$ which we will refer to as $\vdash_{BAINS,\Sigma}$.

The consequence relation $\vdash_{nf,BAINS,\Sigma}$ is sound if $\vdash_{BAINS,\Sigma}$ is sound and $\vdash_{nf,\Sigma}$ is sound and complete if $\vdash_{BAINS,\Sigma}$ is sound and complete.

Definition 2.43 *Assume that* $\vdash_{BAINS,\Sigma}: \mathcal{P}(|Sen_{BAINS}(\Sigma)|) \times |Sen_{BAINS}(\Sigma)|$ *where* $BAINS$ *is a behavioural algebraic institution and* $\Sigma \in |AlgSig|$. *The consequence relation* $\vdash_{nf,BASLN,\Sigma}: SPEX(BASLN) \times |Sen_{BAINS}(\Sigma)|$ *where* $BASLN$ *is an arbitrary but fixed* $BASLnf$ *specification language is defined as follows:*

$$SP \vdash_{nf,BASLN,\Sigma} \phi \Leftrightarrow \Psi \vdash_{BAINS,\Sigma'} \phi$$

where $nf(SP) = <\Sigma', \Psi> |_{\Sigma}$.

Theorem 2.44 $\vdash_{nf,BASLN,\Sigma}$ *is sound if* $\vdash_{BAINS,\Sigma}$ *is sound and* $\vdash_{nf,BASLN,\Sigma}$ *is sound and complete if* $\vdash_{BAINS,\Sigma}$ *is sound and complete.*

3 Compositional proof systems

In this section we present the compositional proof system Π^{SASL}_{AINS} for an $ASLker$ language which we will denote as $SASL$ with a fixed but arbitrary algebraic institution $AINS$. $SASL$ includes just a renaming operator appart from the common operators of $ASLker$ languages and see [BCH] or [Hen97] for an extension of the compositional proof system to observability operators.

Definition 3.1 *SASL is a ASLker language with additionally the following operators:*

$$Signature(rename \quad SP \quad \textbf{by} \quad \sigma) = \Sigma$$

$$Symbols(rename \quad SP \quad \textbf{by} \quad \sigma) = \sigma'(Symbols(SP))$$

$$Models(rename \quad SP \quad \textbf{by} \quad \sigma) = \{ A \in Alg(\Sigma) | A|_{\sigma} \in Models(SP) \}$$

where $\sigma : Signature(SP) \rightarrow \Sigma$ *is a bijective signature morphism and* σ' *is the pushout morphism of the following diagram:*

$$\begin{array}{ccc} Sym(SP) & \xrightarrow{\sigma'} & PO(i,\sigma) \\ {\scriptstyle i}\uparrow & & \uparrow \\ Sign(SP) & \xrightarrow{\sigma} & \Sigma \end{array}$$

where i *is the inclusion* $i : Signature(SP) \hookrightarrow Symbols(SP)$.

The proof system has just the sequent $SP \vdash_{\Pi^{SASL}_{AINS}} \phi$ where $SP \in SPEX(SASL)$ and $\phi \in Sen_{AINS}(Signature(SP))$ and uses the consequence relation $\vdash_{\Pi_{AINS}}: \mathcal{P}(|Sen_{AINS}(\Sigma)|) \times |Sen_{AINS}(Signature(SP))|$ It is defined by the following set of rules:

$$\{ (basic_i) \quad \overline{<\Sigma, \{\phi_1, \ldots, \phi_n\}> \vdash_{\Pi^{SASL}_{AINS}} \phi_i} \mid \phi_i \in <\phi_1, \ldots, \phi_n> \} \cup$$

$$\{\ (pi) \qquad \frac{SP \vdash_{\Pi^{SASL}_{AINS}} \phi_1 \quad \ldots \quad SP \vdash_{\Pi^{SASL}_{AINS}} \phi_n \quad \{\phi_1, \ldots, \phi_n\} \vdash_{\Pi_{AINS}} \phi}{SP \vdash_{\Pi^{SASL}_{AINS}} \phi},$$

$$(suml) \qquad \frac{SP \vdash_{\Pi^{SASL}_{AINS}} \phi}{SP +_\Sigma SP' \vdash_{\Pi^{SASL}_{AINS}} \phi},$$

$$(sumr) \qquad \frac{SP' \vdash_{\Pi^{SASL}_{AINS}} \phi}{SP +_\Sigma SP' \vdash_{\Pi^{SASL}_{AINS}} \phi},$$

$$(exp) \qquad \frac{SP \vdash_{\Pi^{SASL}_{AINS}} \phi}{SP|_\Sigma \vdash_{\Pi^{SASL}_{AINS}} \phi}\ \phi \in Sen_{AINS}(\Sigma),$$

$$(ren) \qquad \frac{SP \vdash_{\Pi^{SASL}_{AINS}} \sigma^{-1}(\phi)}{rename \quad SP \quad \textbf{by} \quad \sigma \vdash_{\Pi^{SASL}_{AINS}} \phi}\}$$

One can prove soundness of the proof system with respect to the satisfaction relation between specifications and sentences and to prove completeness it is needed to assume some properties of the arbitrary but fixed algebraic institution and its associated consequence relation $\vdash_{\Sigma,\Pi_{AINS}}$. See [Cen94] for a proof for the case of a first-order logic and [BCH] for a general proof using institutions.

4 Proof system for refinement

The proof system for the refinement of specifications of the *BASLker* specification language *RBASL* presented in previous section with a fixed but arbitrary algebraic institution *AINS* ($\Pi^{RBASL^{\ggg}}_{AINS}$) has just the sequent $SP \ggg SPI$ for any specification expressions $SP, SPI \in SPEX(RBASL)$ and uses the following definition:

Definition 4.1 *Let ASL be an ASLker specification language with an arbitrary but fixed algebraic institution AINS. Assume that* $SP, SP' \in SPEX(ASL)$. *SP is a persistent extension of SP' (denoted by* $PEXTOF(SP, SP')$*) if the following condition holds:*

- *There exists an inclusion with arity* $Signature(SP') \hookrightarrow Signature(SP)$
- $Models(SP') = Models(SP)|_{Signature(SP')}$.

This proof system is inductively defined by the abstract specification expression as follows:

$$(basic_{\ggg}) \qquad \frac{}{< \Sigma, \Phi > \ggg SPI} \; Signature(SPI) = \Sigma \wedge (SPI \models \Phi)$$

$$(sum_{\ggg}) \qquad \frac{SP' \ggg rename \quad SPI|_{inr(Signature(SP'))} \quad \textbf{by} \quad inrsig^{-1} \qquad SP \ggg rename \quad SPI|_{inl(Signature(SP))} \quad \textbf{by} \quad inlsig^{-1}}{SP +_\Sigma SP' \ggg SPI}$$

$$(export_{\ggg}) \qquad \frac{SP \ggg SPI'}{SP|_\Sigma \ggg SPI} \; Signature(SPI) = \Sigma \wedge PEXTOF(SPI', SPI)$$

$$(reach_{\ggg}) \qquad \frac{SP \ggg SPI}{\textbf{reach} \quad SP \quad \textbf{with} \quad (\mathcal{S}_\mathcal{R}, \mathcal{F}_\mathcal{R}) \ggg SPI} \; Mod(SPI) \models (\mathcal{S}_\mathcal{R}, \mathcal{F}_\mathcal{R})$$

$$(rename_{\ggg}) \qquad \frac{SP \ggg rename \quad SPI \quad \textbf{by} \quad \sigma^{-1}}{rename \quad SP \quad \textbf{by} \quad \sigma \ggg SPI}$$

$$(behaviour_{\ggg}) \qquad \frac{SP \ggg SPI/\approx}{\textbf{behaviour} \quad SP \quad \textbf{wrt} \quad \approx \ggg SPI}$$

$$(abstract_{\ggg}) \qquad \frac{\textbf{behaviour} \quad SP \quad \textbf{wrt} \quad \approx \ggg SPI}{\textbf{abstract} \quad SP \quad \textbf{by} \quad \equiv \ggg SPI} \; Behc(SP)$$

$$(quotient_{\ggg}) \qquad \frac{SP \ggg SPI'}{SP/\approx \ggg SPI} \; Cond(SP, SPI, SPI')$$

where

$$Cond(SP, SPI, SPI') = (Signature(SPI) = Signature(SP) \wedge$$

$$Mod(SPI'/\approx) = Mod(SPI))$$

$$Behc(SP) = Models(SP) \subseteq Models(\textbf{behaviour} \quad SP \quad \textbf{wrt} \quad \approx)$$

and

$$inl : Signature(SP) \rightarrow Signature(SP) +_\Sigma Signature(SP')$$

and

$$inr : Signature(SP') \rightarrow Signature(SP) +_\Sigma Signature(SP')$$

are the pushouts morphisms of $i : \Sigma \hookrightarrow Signature(SP)$ and $i' : \Sigma \hookrightarrow Signature(SP')$, $inl(Signature(SP)), inr(Signature(SP'))$ are the obvious subsignatures of $Signature(SP) +_\Sigma Signature(SP')$ and $inlsign : Signature(SP) \rightarrow inl(Signature(SP))$ and $inrsign : Signature(SP') \rightarrow inr(Signature(SP'))$ are the obvious signature morphisms defined with the pushouts morphisms inl and inr. The proof of the following theorem can be found in [BCH] and in [Hen97].

Theorem 4.2 *For any specification expressions $SP, SPI \in SPEX(RBASL)$, $SP \rightsquigarrow SPI$ if and only of if there exists a derivation of the sequent $SP \ggg SPI$ in $\Delta_{\Pi^{RBASL}_{AINS}\ggg}(SP \ggg SPI)$*

References

[BCH] Michel Bidoit, María Victoria Cengarle, and Rolf Hennicker. Proof systems for structured specifications and their refinements. Chapter 11 of the book Algebraic Foundations of Systems Specification.

[Cen94] María Victoria Cengarle. *Formal Specifications with Higher-Order Parameterization.* PhD thesis, Ludwig-Maximilians-Universität München, 1994.

[Hen97] Rolf Hennicker. *Structured Specifications with Behavioural Operators: Semantics, Proof Methods and Applications.* Habilitationsschrift, Institut für Informatik, Ludwig-Maximilians-Universität München, June 1997.

[HS96] Martin Hofmann and Donald Sannella. On behavioural abstraction and behavioural satisfaction in higher-order logic. *Theoretical Computer Science*, 167:3–45, 1996.

[HWB97] Rolf Hennicker, Martin Wirsing, and Michel Bidoit. Proof systems for structured specifications with observability operators. *Theoretical Computer Science*, 173, February 1997.

[MS85] D. MacQueen and D. Sannella. Completeness of proof systems for equational specifications. *IEEE Transactions on Software Engineering*, SE-11(454–461), 1985.